# Perceptually realistic flower generation

Lu, Zhaoying

*Award date:*
2001

*Awarding institution:*
University of Bath

# PERCEPTUALLY REALISTIC FLOWER GENERATION

submitted by

**Zhaoying Lu**

for the degree of Doctor of Philosophy

of the

University of Bath

2001





ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

To my parents

谨以此论文献给
我至爱的父亲和母亲

# SUMMARY

This thesis describes a method for generating flower growth animation in which a petal surface and shape can be evolved in real time. Most plant modelling currently animates the plant development process by assuming a time interval and the corresponding growth direction, and cannot easily change the time step or deform the shape. This thesis presents a free-form model for surface deformation and growth. The novelty of the method proposed here is the use of a general framework that combines a mass-spring model with control points embedded in a bicubic patch formulation and the integration of the growth function with force control. The key component of this method is to enable the control points to move such that we achieve satisfactory results when the surface growth is constrained by an evolutionary formulation. The evolution theory should take account of natural and artificial perturbations in the growth cycle. This implies that the transient of the control points must be flexible and adaptable. In the model presented here we use a graphical representation for plant growth function, along with a new description of growth force control, to enable the user to obtain flexible parameters for surface control. In addition, genetic algorithm techniques have been used to optimise the collision avoidance among the flower organs and the environments. The model generates non-deterministic evolutionary results which give more realistic and varied growth than can be obtained using pre-defined surfaces or interpolating between given initial and final shapes.

# ACKNOWLEDGEMENTS

I wish to express my thanks to my supervisors, Dr. Claire Willis and Dr. Derek Paddon, for their invaluable advice and guidance throughout this research and in producing this thesis.

I would like to thank all the former and present members of the department for their support and many interesting discussions during my research.

Thanks also due to University of Bath for its research studentship.

Finally, special thanks to my parents, my sister and my best friend Teck for their encouragement and continuous support during the last five years.

# Contents

# List of Figures

# Chapter 1

# Introduction

The use of computer graphics method to simulate and animate biological entities has been continuously developed since computers became available to scientists in the second half of the last century. One of the early methods to animate mammalian life forms was the key frame method. Later, fractals were used to generate images of objects as diverse as landmarks and the human face. Early methods for the animation of plant forms were developed by Prusinkiewicz [Prusi90a]. Here L-systems give fractal like images for branching structures such as trees.

The animation of flowers has been neglected, other than very simplistic methods that use libraries of pre-defined shapes and components for the user to build into completed entities. Surface shapes in flower components: petals, sepals, stamens and carpels can vary from simple to very complex. But the most difficult problem to solve in flower animation is not a single component shape, but its evolution over its life-cycle from its embryonic state to its fully developed state and then its transformation to a seed bearer.

This thesis describes a method for generating flower growth animation in which a petal surface and shape can be evolved in real time. Most plant modelling

systems currently animate the plant development process by assuming a time interval and a corresponding growth direction. Unfortunately, this method is restrictive in that one cannot easily change the time step or deform the shape. This thesis presents a free-form model for surface deformation and growth. The novelty of this method, proposed here, is the use of a general framework that combines a mass-spring model with control points embedded in a bicubic patch formulation. The key component of this method is to enable the control points to move, such that we achieve satisfactory results when the surface growth is constrained by an evolutionary formulation. The evolution theory must take account of natural and artificial perturbations in the growth cycle. This implies that the transient movement of the control points must be flexible and adaptable, thus the model generates non-deterministic results which give more realistic and varied petals than can be obtained by using pre-defined surfaces or interpolating between given initial and final shapes.

## 1.1 Background

When we build virtual environments, it is relatively easy to generate walls, desks, chairs, machines and other man-made objects because their shapes are designed by ourselves. However, it is quite hard to simulate biological processes because they grow and move according to highly complex natural principles, which can be very difficult to model. Of course, if we restrict ourselves to deterministic models of growth, we can find equations which will allow us to generate the final model of an organism. However, this approach will fail to provide the natural growth of organisms and miss the infinite perturbations found in nature and thus lose not only realism but also much of the natural beauty found in nature. In this work

we strive to find a non-deterministic method that accounts for natural growth.

The application we are interested in involves the animation of flower development. This thesis focuses on the simulation of flower petals through all transient stages, from a bud to the fully developed flower. Thus, we need to develop methodologies that support this simulation, and we must also ensure that we achieve realistic results of all time transient behaviours.

Smooth surfaces are of great importance in geometric modelling and computer graphics. Classic mesh methods are widely used to model complex smooth surfaces such as those encountered in cloth and human character animation. However, these methods suffer from at least two drawbacks. First, calculation is expensive when the number of mesh nodes is very large. Second, it is difficult to maintain smoothness, especially when mesh nodes need to be repositioned as the model builder requires. Bicubic patches have the potential to overcome both of these problems: they have limited control points, and smoothness of the model is automatically guaranteed, even as the model animates. Bicubic patches, among other methods will be explored to judge if they offer the properties we require.

Simulation of surface deformation and growth is a key challenge in virtual environment animation. A realistic interactive surface model has been developed to make it possible to manipulate and control such surfaces. It is usual to generate surfaces using mathematical representations based upon polynomial functions of two parameters, such as Bézier surfaces, B-splines or rational B-splines. Such surfaces can be defined by an array of control points. However, it is a difficult task to enable the control points to move in such a way that we create a desired shape in an interactive environment. Since it is necessary to obtain a growing surface for representing petals or leaves, it is important to construct a suitable dynamic surface model. In this research, we will explore the combination of the control

forces with the mass-spring model on the surface control points. These control forces are also related to the growth force. We should evaluate the necessary forces to enable the petal surface to grow to the desired shape.

Simulation of collisions between mature organs is also an important problem in the visualisation of structures with densely packed organs such as flowers. In nature, individual flowers touch each other, which modifies their positions and shape. Consequently, the mature organs should be carefully modelled and sized to avoid intersections. This is a feasible goal if modelling static structures, but proper simulation of collisions becomes difficult in the realistic animation of plant development. But it is crucial we achieve this simulation. We will investigate the incorporation of genetic algorithms for optimising the moving space for the control points, with the objective of avoiding collisions while the petals are growing.

## 1.2 Thesis Objectives and Overview

The objectives of this thesis are to:

- Present an interactive surface deformation and growth model, the purpose of which is to control the surface and meet perturbations in the growth period.
- Present the integration of growth theory and the control points forces model, with the mass-spring framework, to enable the surface to grow according to biological growth theory.
- Illustrate step-by-step, the control for surface development with an analysis of the shape changes.

- Describe the shape control by the implementation of a growth function, which gives a good description of surface development.

- Investigate a genetic algorithm method for controlling surface collision and deformation.

Firstly, we describe the difficulties associated with surface modelling and growth, and explain why these difficulties can be solved by the model presented here. Secondly, we introduce a mass-spring model and the constraints on the deformation rates of the springs, and the growth rates in order to show the force effect. Thus a petal is approximated by a deformable surface composed of a network of masses and springs, the movement of which evolves using the numerical integration of the fundamental law of dynamics and growth theory. We take these constraints into account using a low-cost method to implement the growth model.

We demonstrate that the combination of bicubic patches and the mass-spring model is an appropriate and efficient interface to physical simulation for animation, and the availability of a growth function enables our surface model to be used as an extremely effective tool in developing virtual environments. We will show that the introduction of growth functions and dynamic forces gives the resulting surface the appropriate motion and physical properties of a real petal.

Finally, we develop a genetic algorithm technique to optimise the collision avoidance among the flower organs and the environments. The essence of the method to be developed uses fitness function to meet the natural growth criteria.

## 1.3 Thesis Structure

Here we use a diagram (Figure 1.1) to show all the components presented in this thesis and their relationships.

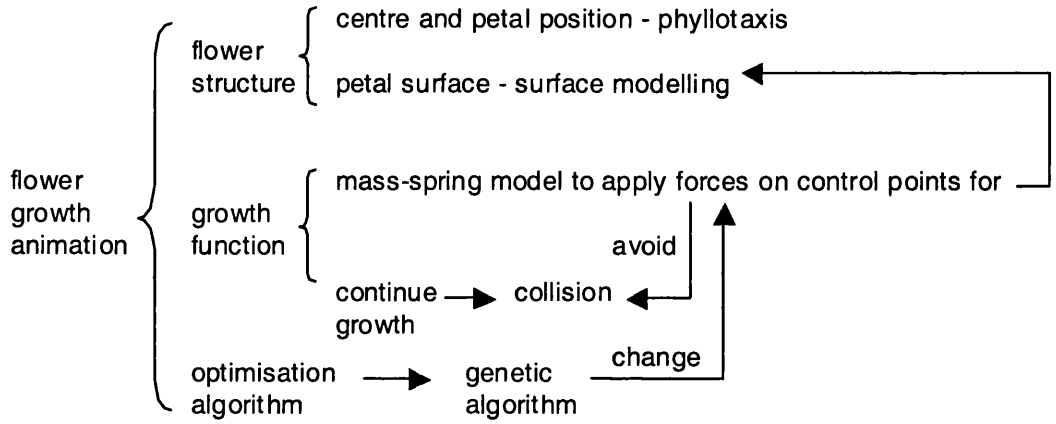



Figure 1.1: The thesis structure and relationships

Our flower growth animation consists three areas: flower structure, growth function and optimisation algorithm. These three areas have developed different contents. These components are related to each other. And the influences between them enable the flower to have realistic growth.

Initially we concentrate on flower structure, which consists of the flower centre and petal placement which is implemented with a phyllotaxis method and petal surface model. The second part is growth theory. We apply the growth function with a mass-spring model to control the petal surface growth. The third part in-

volves an optimisation algorithm, which helps to implement the flower interactive growth. At this point in the flower development we must consider the methods to avoid petal intersecting with each other, and also neighbouring flowers must not intersect. Here, we apply collision avoidance method, introduced with a genetic algorithm.

## 1.4 Layout of the Thesis

The outline of the thesis is:

**Chapter 2** gives the graphic background for the work in this thesis. The main terms and definitions of the subject area are described. It investigates the appropriate method for representing the fundamental surface model.The chapter also reviews the concept and importance of perceptually realistic generation for the developed model. Also we review the related work from free-form surface modelling for evolving surfaces, and the limitation and drawbacks from those techniques.

**Chapter 3** reviews the theoretical background of phyllotaxis and presents the phyllotaxis structure for plant components. It also develops the spiral phyllotaxis structure for flower centre and petals, and illustrates the position placement method for the flower model.

**Chapter 4** begins with an analysis of existing surface model and develops the surface representation for our flower petal model. It also presents the theory of flower petal shape, the purpose of which is to give an accurate description for the shape of plant components. It also illustrates step by step the petal development controls with analysis of the shape changes.

**Chapter 5** presents a theory of growth functions, the purpose of which is to give a good description of plant development. Then we develop a graphical representation for plant growth functions and introduce the integral equations for growth measurement, based on a specific time interval. The chapter concludes with the description of the shape control required by the growth function.

**Chapter 6** extends the ideas of mass-spring model and present our improved structure for controlling growth. We develop an integration of growth theory with both the control points forces model and the mass-spring framework, to enable the surface to grow according to biological growth theory. We also present the appropriate collision detection and collision response methods for an evolving petal surface model.

**Chapter 7** reviews the theoretical background of genetic algorithm and presents reasons for choosing this method for collision avoidance. We develop a fitness function for model. We end this chapter with a crucial review of what the use of genetic algorithm has achieved.

**Chapter 8** presents the conclusions of the thesis and the recommendations for further research.

**Appendix A** reviews the whole implementation for flower growth generation. We discuss all the parameters and factors used in the system user interface, and review the other relevant algorithms and techniques applied in the animation system.

# Chapter 2

# Background Theory

## 2.1 Introduction

In this chapter we introduce some important background materials which support much of the work in this thesis. We also present the motivation for doing this research and the goal of the presented model. We begin with defining the surface theory and conclude with the concept of perceptually realistic surface generation.

In addition, we review the related work in the area of surface modelling, such as B-spline surfaces, meshes and subdivision surfaces. We also examine the similarity of various techniques for surface growth.

## 2.2 Goal of the Model

The goal of this thesis is to investigate theories and their implementation that allow a user to specify and animate a flower through its complete life-cycle. This is essentially the modelling of a free form surface.

Although there are aesthetic views associated with the animation of a flower,

and particularly its petals, there still remains at the heart of this problem the need to represent a surface and its evolution. Therefore, we must first consider representation of surfaces and judge their relative merit in the context of biological modelling. We start by considering the approximation of lines and their generalisation to surfaces.

### 2.2.1 Curves

In computer graphics we can use several approaches for representing curves. One is similar in concept to the polygonal approximation technique. It approximates a curve through a series of straight lines and is consequently known as piecewise linear approximation. This has the advantage of being extremely simple conceptually as its representation is simply a series of points. However, it has several disadvantages: the first being that it can be very difficult to edit the shape of a curve, it may involve selecting each of many points and moving each point individually; the second disadvantage is that the number of points required to produce a good approximation of a curve may produce very large quantities of data, which may in turn slow down the interaction with the program; a final criticism is that the curve is never truly smooth.

In view of the above criticism, most computer modelling systems provide another technique which makes uses of splines to represent curves [Foley90]. One of two broad categories of splines is the interpolating spline. In this form, the spline curve passes directly through each of the control points. The obvious disadvantage is that this direct relationship makes it difficult to generate curves having a very smooth and gradual curvature. The second basic category of spline is the approximating spline. It places the control points by calculating the curve

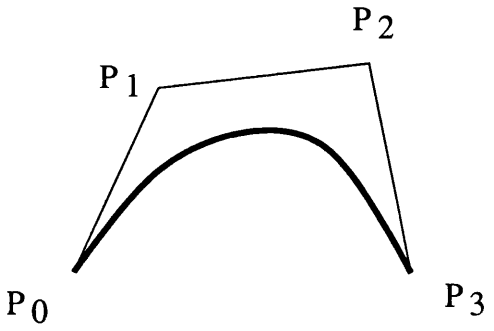



Figure 2.1: A Bézier curve and its control points

such that the curve goes near the control points, but not directly through all the control points. One of the most commonly used approximations is the B-spline, that is the Basic-spline. we should notice that the B-spline curve does not actually touch any of the control points. The Bézier spline is another very popular variety of approximating spline. A basic cubic Bézier spline segment is defined by four control points (see figure 2.1). There are a number of advantages for using parametric descriptions in computer graphics rather than implicit functions. As a particular form of parametric representation, the curve is usually specified by:

$$P(u) = \sum_{i=0}^{3} P_i B_{i,3}(u) \tag{2.1}$$

where each term in the sum is a product of a control point $P_i$ and a blending function $B_i$ which in this case is a polynomial of degree three. These curves show the influence that each control point has on the final curve form [Farin88a].

### 2.2.2 Biparametric cubic surfaces

The treatment of parametric cubic curve segments, given in the foregoing section, is easily generalised to bi-parametric cubic surface patches. A point on the surface patch is given by a bi-parametric function and a set of blending or basis functions

are used for each parameter. A cubic Bézier patch is defined as:

$$P(u,v) = \sum_{i=0}^{3}\sum_{j=0}^{3} P_{ij}B_{i,3}(u)B_{j,3}(v) \tag{2.2}$$

Mathematically these three-dimensional surfaces are said to be generated from the Cartesian product of two curves. More technically, since it is formed from two cubic splines, it is called a bicubic patch, "bi" meaning "two" and "cubic" referring to the fact that the mathematics of a spline curve involve powers of three. On a bicubic Bézier patch (see figure 2.2), its sixteen control points bear a relationship to the shape of the surface, in the same way that the characteristic polygon relates to a curve segment. It can be seen intuitively that twelve of the control points are associated with the boundary edges of the patch (four of them, that is $P_{00}, P_{03}, P_{33}, P_{30}$, specifying the end-points) and the four interior points, that is $P_{11}, P_{12}, P_{22}, P_{21}$, specify the internal shape. Only the corner points lie in the surface. The properties of the Bézier curve formulation are extended into the surface domain. A single displaced control point can be used to easily deform the whole patch surface. The intuitive feel for the surface through its control points and the ability to ensure first-order continuity are maintained. (The first-order continuity means that the tangent vectors at the end of one curve and the start of the other match to within a constant, which means the smoothness of the curve is maintained at the joint point.) The surface patch is transformed by applying transformations to each of the control points.

The way in which the control points work can be seen by analogy with the cubic curve. The geometric interpretation is naturally more difficult than that for the curve and, of course, the purpose of the Bézier formulation is to protect the designer against having to manipulate tangent vectors.

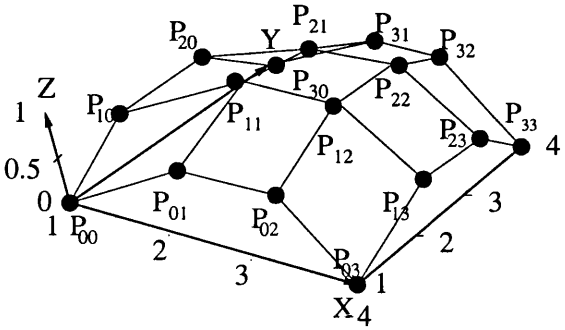



Figure 2.2: Control points for bicubic Bézier patch

### 2.2.3 Parametric representation comparison

B-spline surfaces patches are related to B-spline curves in the same way that Bézier patches are related to Bézier curves. The advantages of B-splines over Bézier curves are possibly more important in the surface domain. The non-localness of the control point influence in Bézier curves is a disadvantage in deforming or modelling contexts and the joining condition for multipatch segments imposes tedious constraints and makes user interfaces difficult. The non-localness implies that, although a control point heavily influences that part of the curve most close to it, it also has some effect on the whole curve. However, when we consider the growth of the flower petal surface, it shows that the growth happens over the whole surface at the same time. So the surface does not change locally and does not require the advantage of B-spline surface. Another of the major disadvantages of the sixteen-point Bézier patch is that it can only represent a surface element that describes a simple topography. In contrast, a B-spline surface has no limit placed on the number of points defining the characteristic polyhedron, and therefore it is possible to represent far more complex surfaces using a single B-spline patch compared to a single Bézier patch. However, again difficulties may arise for a user interface controlling object deformation or modelling as the

number of control points increases, resulting in more complexity for the user to manage.

A more general problem arises when modelling with B-spline surface patches. Because a patch does not interpolate or pass through its control points there is a difficulty in obtaining an initial surface approximation, from say a network of modelling points, for use in subsequent deformation. This approximation has to be made by an initial process which thereby yields a set of control points.

Two important surface representation schemes exist that extend the control of shape deformation beyond that induced by the movement of control points. These are NURBS (non-uniform rational B-splines) and $\beta$-splines. Both these curve approximation methods can be extended to tensor product surfaces in the same way that the Bézier and B-spline curve definitions were used as a surface basis. A singular disadvantage is that $\beta$-spline curves do not pass through any control points (including the end-points). NURBS provide difficulties for the user as they introduce many control points.

### 2.2.4 Shape description requirements

Generally, shape representations have two different uses, an analytic use and a synthetic use. Representations are used analytically to describe shapes that can be measured; just as a curve can be fitted to a set of data points, a surface can be fitted to the measured properties of some real objects. The objective of such representations may be to achieve a precise fit, to minimise the number of measurements required, to represent the shape in a very compact form, to simplify the computation of derived properties such as areas and volumes, etc. Synthetic uses of shape representation are encountered in design. A designer interactively

creates or modifies a model of a shape, examining and improving the design until it is acceptable. The objectives for synthetic uses of shape differ from analytic objectives: we are primarily concerned with expressing shape modifications easily in an interactive program, with the freedom to explore many very different alternative shapes.

In this thesis, our treatment of the mathematical techniques applied to the modelling of shape will concentrate on the synthetic approach. In this approach we require a clear understanding of the user's needs and the constraints of the application.

## 2.3 Related Work

The related work is in two main parts: the first part discusses related work in the area of surface modelling; the second part discusses models for the growth of surface features. We also present the limitations and restrictions of these modelling techniques.

### 2.3.1 Free form surface modelling

Free-form surfaces are proven to be important in computer-aided geometric modelling and are met in many cases of practice. The goal for interactive modelling of free-form surfaces is to make it easy for the user to control the shape of the surface.

Recently, many attempts have been made to manipulate surfaces. Some interesting results have been obtained using implicit equations [Seder95], mathematical frameworks [Skala97], convex parametric surface patch fitting [Juttl98], partial differential equations [Ugail99], and boundary element methods [Doug99].

However, all of these methods require many surface details and are not easily controlled. The interactive design technique we defined here should allow the user to manipulate the surface effectively with a minimal number of control points.

Classic mesh methods are widely used to model complex smooth surfaces such as those encountered in cloth and human character animation [Provo95]. There are some differences between the nature of these two kinds of surface. The cloth is elastic, stretchable and distortable. However, a natural plant organ, such as a flower petal, is distortable, but not elastic, or stretchable. In addition, a plant organ is stiffer than cloth. The most important feature of a plant organ is that it grows, which is not equivalent to stretching. Growth involves size change, shape change and mass change, and maybe even distortions if there is any external intervention or any obstruction. It is therefore essential that our model is sufficiently rich to enable such growth to be animated.

The collision and deformation of a free-form surface model remains a challenging area because of the controlling flexibility of the surface. Therefore, we need to choose a surface model with easy access to collision detection and deformation. Let us look at the related work in the collision detection area. The algorithm in [Krish97] formulates the intersection problem algebraically, computes the projection of the intersection curve as an algebraic plane curve, and evaluates it. The further detection involves the efficient computation of surface intersection. It proposes a matrix representation for the intersection curve and computes it accurately using matrix computations. It computes a start point on each component of the intersection curve, detects the presence of singularities, and finds all the curve branches near the singularity. The idea can be applied in our detection method. However, more efficient algorithms must be developed for interactive modelling.

### 2.3.2 Growing surfaces

The other important requirement for our surface model is that it can grow. There are several related growing surface models. One major area of interest is for animals and its organs, the other is for plant growth.

Durikovic describes a growth model for the stomach. He presents the shape of the organ by a number of ellipsoidal clusters centred at points on the skeleton [Durik98a]. He also introduces several tables with which to store the database of statistical geometry of organs, such as size, growth speed, among others. The method simply uses the model to generate the shape from the stored data, thus it does not provide natural growth control. This method is therefore unsuitable for a continuous growth model.

Recently, many attempts have been made to simulate the development of plants, trees and botanical structures. Some interesting results have been obtained using branching process constructions [Aono84a], particle systems [Demko85a], ramification matrix of trees [Vienn89a] and strand internal vascular structure on trees [Holto94a, Weber95a]. A virtual plant system has been generated by parametric L-systems, which is a recursive technique, and can model highly complex and irregular structures [Prusi90a]. Mech built a modelling framework to simulate and visualise a wide range of interactions at the level of plant architecture [Mech96a]. Fowler uses spiral phyllotaxis to model flowers [Fowle92a]. However, most of this work has focused on a target structure and ignored the transient stages of plant growth and development. The main problem with these systems is that they do not support level-of-scale simulation. In the main, zooming in would expose poor images. If we assume that the user is in complete control of level-of-scale operation, then we must carefully model the surface features of

the plant and all transient stages during the development of the plant. Level-of-scale operations are increasingly important in virtual environments, therefore we should reject any method which is unsuitable for this requirement.

## 2.4 Shape Representation

### 2.4.1 Major points for choosing the modelling and representational method

Many factors need to be considered when choosing the right representational method for the model. These are:

- The generality of the representation. What are the limitations on the type of three-dimensional shape that can be represented?
- The relative difficulty in building or specifying the initial structure.
- The data storage requirements.
- The ability of the method to accommodate an interactive editing scheme.
- The ease of rendering the object.
- The generality of the method with respect to applications.

### 2.4.2 Shape representation

From the related work described in the last section, we found drawbacks and limitations from those techniques implemented in their work and concluded that they are unsuitable for our flower petal surface modelling. With the criteria

described above, there is an alternative technique for our petal surface. It is the bicubic parametric patch. We present the reasoning behind this choice below.

A real object (or a physical model of a real object) can be represented by a net or mesh of patches, but the representation may not be exact. It is possible to model subtly shaped objects such as a human face with a net of patches. An adequate representation of such an object using a polygon mesh would need an extremely high polygonal resolution. The real value of the representation here is that it can be used to transform an abstract design, built up within an interactive program, directly into a physical reality. The description can be made to control the petal surface growth without any human intervention. It is this single factor more than any other that makes bicubic parametric patches important.

The apparent advantages of this representation over the polygon mesh representation are:

- It is an exact analytical representation;
- It has the potential of three-dimensional shape editing;
- It is a more economical representation.

To match all the factors considered in the last section when choosing the right representational method for our model, we make the following comments:

- A bicubic patch with sixteen control points can only represent fairly simple surfaces. However, we will show this is sufficient for our flower petal application in the area of perceptually realistic modelling.
- The way we position the control points on the bicubic Bézier patch provides easy access to the structure. The deformation of any part of the structure can easily be introduced by using the relevant control points.

- It is obvious that bicubic patches are very economical to store as only sixteen control points are used.
- By modifying the standard structural model with force control and mass-spring nodes, we are able to provide much better user control.
- Bicubic patches are easily rendered with ray tracing.
- With the particular growth feature for our surface model, the bicubic patch representation method will be an efficient way for our application.

Given these advantages it is somewhat surprising that this form is not the mainstream representation in computer graphics. It is certainly no more difficult to render an object represented by a net of patches and so we must conclude that perhaps its lack of popularity in mainstream computer graphics is due to the mathematical formalisms associated with it. However, with our improved structure modelling and biological force control, bicubic patches are shown to be a wise choice for our application. The details of the petal surface modelling will be discussed in the later chapters.

## 2.5 Perceptually Realistic Modelling

To understand our model completely requires knowing some general concepts for visual perception [Sekul94a] and the ways all the constituent components needed by our model interact with each other. A complete understanding of perception must include a thorough description of the appearances of objects or events. To describe how things appear to us, we must specify how our senses detect and recognise objects.

The aim of an interactive animation system is to create an exciting and real experience for viewers, to give them a feeling of immersion, of "being there". A visual experience must be created for them that mimics the events that happen in their real environment, or that matches their expectations of what might happen in an environment that they have never experienced (e.g. in space, or a high-speed racing car, or a virtual world that may not exist in the real world).

In this section, we build up the requirements for the generation of plant components' structure and their growth models. Because we are interested in perceptually realistic models, we do not necessarily have to specify the requirements of that found in the real world. Measure of scale may also be taken into account in this specification.

### 2.5.1 Perceptually realistic generation

We are viewing perception as a dynamic process in which sensory messages play an integral but subordinate part. From visual perception theory [Bruce96a], we can build up the ideas about accuracy requirements for our generation model.

Firstly, human perceptions tends to perceive the shape and size of an object as constant, even when it is viewed from a different position, or when the object has changed slightly.

Secondly, humans perceive a moving object less well than a static object. The faster the object moves the less detail we can perceive. It means that we cannot catch very fast movement. Thus, we must choose a suitable time step for the growth animation in our model.

Thirdly, humans are better at perceiving changes in the continuous part of an object, than changes at discontinuity points. Thus we must ensure growth model

can always generate a smooth surface.

Finally, we find it easier to perceive three dimensional objects with complex, asymmetrical and discontinuity features. So, in a group of 3D objects, we will perceive the one that is changing in a more complex fashion, faster than the ones changing in a simple fashion.

From the above relevant analysis for our visual perception, we can see that the acuity criteria for our model can be adjusted to match the specific computer simulation that we require. It provides the possibility, in some cases, to have realistic plant generation without considering the biology details.

It is known that the biological control mechanisms for plant growth are complicated and cannot in general be formulated as a mathematical model. Therefore, it is unlikely that we can find a formulation that will enable us to simulate all the biological detail found in flower growth. Instead, we concentrate on finding simple methods, mathematical or empirical, that will give us plant simulations that are perceptually realistic. As surface changes in a flower are the most readily perceived by a typical observer, we will concentrate our research on achieving realism in surface changes that occur during the growth cycle of a flower.

Perception is not the passive processing of information through specific channels; obviously it is an active process. Other models of perception—particularly those based on a computer simulation of the coding and encoding of input—have often failed to acknowledge the active role of the perceiver. We present the perceptual model with the lower acuity requirements for a smooth surface growth changing by the analysis from the visual perception system, which tends to perceive constant, slow, continuity and good shape characteristics.

## 2.6 Conclusions

In this chapter, we have presented the goal of our model and the model design requirements for the user. With the illustrations of the curve and surface theory, we analyse and compare the advantages and disadvantages of the representation methods for surfaces. We emphasised the need for these surfaces to represent static flower models, and to be able to simulate a biological life-cycle for the plant.

From the considerations of mathematical, computational and complexity analysis, we have chosen bicubic Bézier patches for our surface model. The reasons are listed that show this choice is an efficient method for our application. In later chapters we will develop these arguments to demonstrate the utility of our choice.

Our reviews of related work have suggested that insufficient research has been previously undertaken in the flower growth animation for virtual environments. From the natural and biological standard point, existing techniques, such as L-systems, are inappropriate if the requirement is realistic biological animation. We must conclude that further research is needed to enable detailed animation of plant forms to be achieved.

We have initiated the discussion for the goal of the perceptually realistic model. By introducing certain structural cues, we have indicated that surface representations are an important component in the desire to achieve overall plant life-cycle events.

# Chapter 3

# Modelling for the Spiral Phyllotaxis

## 3.1 Introduction

This thesis focuses on the simulation of flower growth through all transient stages from a bud to the fully developed flower. Thus, we need to develop methodologies that support this simulation, to ensure that we achieve realistic results for time transient behaviours. In this chapter we introduce some important biological background materials which support the growth stages described in this thesis. Most of what we describe are related to flower centre and petals position arrangements.

A number of models have been introduced over the past few decades for producing realistic images of organic structures such as trees, flowers, fruits and shells. Parametric L-systems are recursive algorithms that can model highly complex and irregular structures such as trees, while collision-based spiral phyllotaxis systems [Fowle92a] can model complex but regular structures, such as the

placement of petals and seeds on a flower head.

In fact, we live in a universe of patterns, just like no two snowflakes are ever exactly the same, but they all have sixfold symmetry. By using mathematics to organise and systematise the rules of natural processes, we can build the best models for interactive virtual environments.

Plant organs are often arranged in spiral patterns. This effect is termed spiral phyllotaxis. Well known examples include the layout of seeds in a sunflower head and the arrangement of scales on a pineapple. In this chapter, we begin by defining the phyllotaxis theory for plant components and conclude with the application of these methods to the growth development of flower centres and petals that we simulate in this thesis.

The arrangement of leaves, petals, and other organs in plants has a large and distinguished literature. But early approaches to plant development are purely descriptive, they do not explain the physical characteristics of plant growth, they just describe the geometry of the arrangements of the plant organs. In the last ten years, some graphics researchers started to simulate plant development with different models. The collision-based model [Fowle92a] describes distribution of flower initial parts, or primordia, on a supporting surface, called the receptacle, which determines the shape of the entire structure. Although the model operates correctly for various combinations of receptacle shapes and primordia sizes occurring in nature, it does not provide ready-to-use formulae relating the arrangement of spirals to the geometry of the receptacle and the sizes of primordia. In nature, individual flowers touch each other, which modifies their positions and shapes. This effect is not captured by the present collision-based model, since collisions are detected only for primordia. Consequently, the mature organs must be carefully modelled and sized to avoid intersections. This is feasible while modelling

static structures, but proper simulation of collisions would become crucial in the realistic animation of plant development. We will extend the application of phyllotaxis from seeds on the flower centre to the flower petals, which helps to arrange the petal in priority order and respond to petal surface collision correctly. We will explain these in detail from the basic phyllotaxis theory to implementations in the whole flower structure.

## 3.2 Basic Phyllotaxis Theory

If you look at the tip of the shoot of a growing plant, you can detect the organs from which all the main features of the plant develop, such as leaves, petals, sepals, florets, or whatever. At the centre of the tip is a circular region of tissue with no special features, called the apex. Around the apex, one by one, tiny lumps form, called primordia (figure 3.1). Each primordium migrates away from the apex – or, more accurately, the apex grows away from the lump – and eventually the lump develops into a leaf, petal, or the like. Moreover, the general arrangement of those features is laid down right at the start, as the primordia form. So basically all we have to do is explain why you see spiral shapes and arrangements that satisfy Fibonacci numbers in the primordia.

The first step is to realise that those spirals most apparent to the eye are not fundamental. The most important spiral is formed by considering the primordia in their order of appearance. Primordia that appear earlier in the formation migrate farther, so you can deduce the order of appearance from the distance away from the apex. What you find is that successive primordia are spaced rather sparsely along a tightly wound spiral, called the generative spiral. The human eye picks out the Fibonacci spirals because they are formed from primordia that

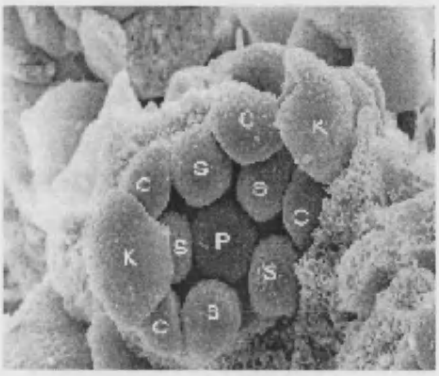



Figure 3.1: Primordia on plant apex

appear near each other in space; but it is the sequence in time that really matters.

Phyllotaxis (alternative, phyllotaxy) is the term applied to the sequence of origin of leaves on a stem. The phyllotaxis of any one plant, or at least any one shoot on a plant, is usually constant and often of recognisable value. The relative positions of leaves on a plant must affect the interception of light and, more importantly, the position of a leaf usually fixes the position of its subtended axillary bud. Thus the phyllotaxis of a plant can play a considerable role in determining the branching pattern of a plant, particularly for woody perennials. The study of phyllotaxis has led to an extensive terminology and also to a preoccupation with the Fibonacci series [Bell91]. This is because most natural phenomena show the relationships between phyllotaxis and Fibonacci series.

### 3.2.1 Spiral phyllotaxis

Phyllotaxis applied to leaves is very complicated. However, it is customary to describe the phyllotaxis of plants having the following patterns as: distichous (figure 3.2); tristichous (figure 3.3); and spiral (figure 3.4 and figure 3.5) in terms of a fraction, i.e.$\frac{1}{2}, \frac{1}{3}, \frac{2}{5}$, etc. This fraction is a measure of the angle around the stem between the points of insertion of any two successive leaves. Thus in

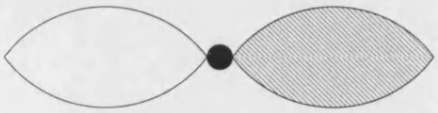


Figure 3.2: Phyllotactic arrangement:distichous

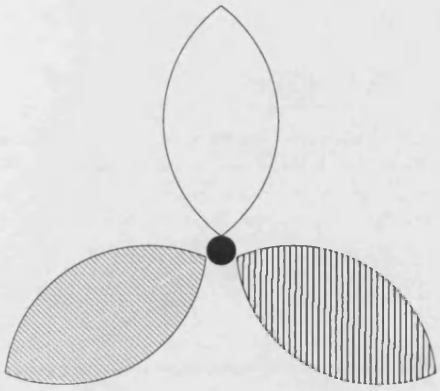


Figure 3.3: Phyllotactic arrangement:tristichous

$\frac{1}{3}$ phyllotaxis (tristichous) there is $\frac{1}{3} * 360° = 120°$ between two longitudinally adjacent leaves, in $\frac{2}{5}$ phyllotaxis there is $\frac{2}{5} * 360° = 144°$ between two successive leaves (figure 3.6). An imaginary line can be drawn spiralling around such a stem which passes through the point of attachment of each next youngest leaf in turn. This is termed the genetic spiral (figure 3.6 and figure 3.7).

An estimate of the phyllotactic fraction can be found by following the genetic spiral around the stem from any one older, lower, leaf to the first younger leaf directly in line above it. Leaves seen to be arranged in a common longitudinal line are said to lie on the same orthostichy. The example of $\frac{2}{5}$ phyllotaxis (figure 3.6) has five orthostichies. In figure 3.6 the lower leaf will be given the number 0 and the leaf arrived at vertically above it will be found to be number 5. The genetic spiral will have been found to have passed twice around the stem giving a fraction of $\frac{2}{5}$ and hence an indirect measure of 144° between any two successive leaves.

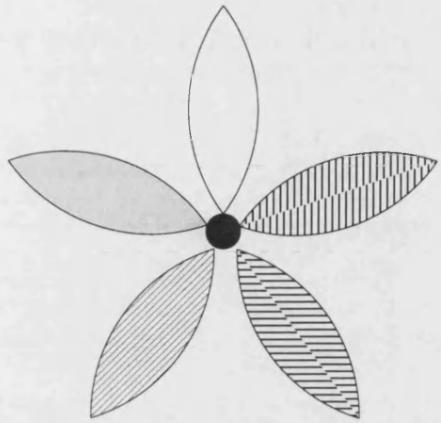


Figure 3.4: Phyllotactic arrangement:spiral

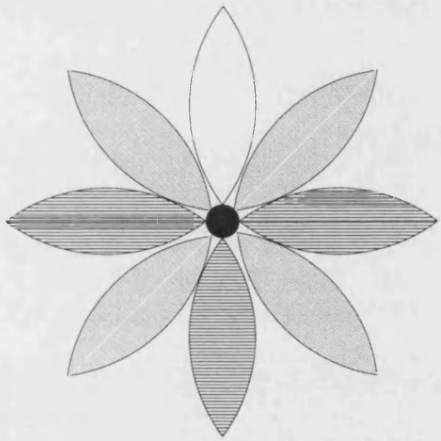


Figure 3.5: Phyllotactic arrangement:spiral

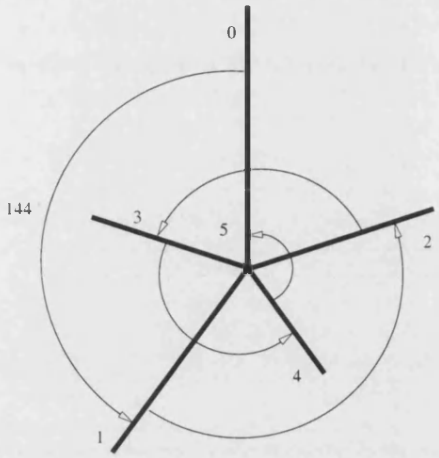



Figure 3.6: $\frac{2}{5}$ phyllotaxis, the arrows follow the genetic spiral, positions 0 and 5 lie on the same orthostichy.

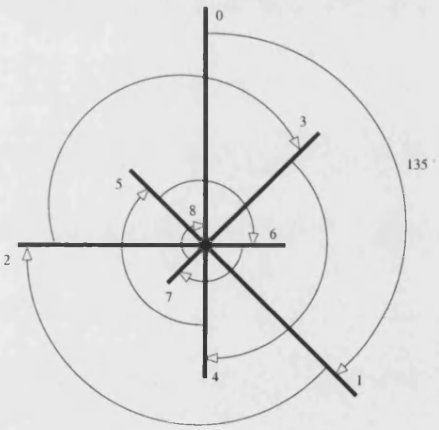



Figure 3.7: $\frac{3}{8}$ phyllotaxis, the arrows follow the genetic spiral, positions 0 and 8 lie on the same orthostichy.

In figure 3.7 the phyllotactic angle is 135° ($\frac{3}{8}$, i.e. leaf 8 is above leaf 0 and reached by passing three times around the stem). The ease with which this measurement can be made may be more or less confused by the amount of internode twisting or leaf primordium displacement that has taken place as developing leaves become shifted away from initially precise orthostichies. These biological theories are also applied to the arrangement of flower petals. So our system has to cope with different angles for different petal numbers. The relationship between the phyllotactic angle and petal number is similar to the fraction described above. More details are followed in the next section.

The phyllotactic fractions almost invariably found in plants with spiral phyllotaxis are:

$\frac{1}{2}$ $\frac{1}{3}$ $\frac{2}{5}$ $\frac{3}{8}$ $\frac{5}{13}$ $\frac{8}{21}$ $\frac{13}{34}$ .....

which represent angles of:

180° 120° 144° 135° 138°28′ 137°6′ 137°39′ .....

### 3.2.2 Fibonacci numbers and petal numbers

The presence of the Fibonacci series and hence the golden ratio in the phyllotaxis of plants has led to much investigation and many explanations. A logarithmic spiral can be extended indefinitely outwards or inwards and is therefore always of the same shape regardless of its dimensions. The shell of a snail forms such a spiral. As the animal increases in size it occupies a progressively larger volume. However, both the animal and its shell retain the same shape regardless of size. A similar growth phenomenon takes place at the apical meristem[1] of a plant when leaf primordia of initially small size develop but of necessity occupy the same

[1]All plant growth originates in localised regions of perpetually embryonic tissues; these regions are called meristems. Meristems located at the tips of all roots and shoots are called apical meristems.

proportion of the apex surface. The consequence of this packing of enlarging organs can be seen on a pineapple fruit or on the inflorescence head of a sunflower. All the sunflower seeds are the same shape but not the same size. Furthermore, they are arranged in radiating spiral rows; two directions of rows are visible, one set clockwise and one set anticlockwise. These rows are termed parastichies and form logarithmic spirals. All the spaces between these intersecting logarithmic spirals are the same shape regardless of size.

Developing leaf primordia enlarging at a growing shoot apex similarly continue to fit comfortably together as they expand in basal area and will inevitably form two sets of interlocking parastichies in the process. This uniformity of shape resulting from logarithmic spirals does not occur unless the number of parastichies in each direction conforms to the Fibonacci series. Thus counts of rows on sunflower heads, or pineapple fruit, conform to the following series: 1, 2, 3, 5, 8, 13, 21, 34 etc. in one direction and 2, 3, 5, 8, 13, 21, 34, 55 etc. in the other direction. Intermediate combinations do not occur and would result in distorted structures. This series is complementary to the series giving a measure of the angle between any two successive leaves on the genetic spiral as it gives a measure of the angle.

It seems that there is some kind of dynamic constraint on plant development, which naturally leads to Fibonacci numbers. That is the reason we take the Fibonacci numbers as the possible petal numbers and set their proper phyllotactic angels in the implementation. However, the user would have an asymmetrical flower if he requires an odd number of petals, such as seven. In the following sections, we start to describe the spiral model implemented on the flower head.

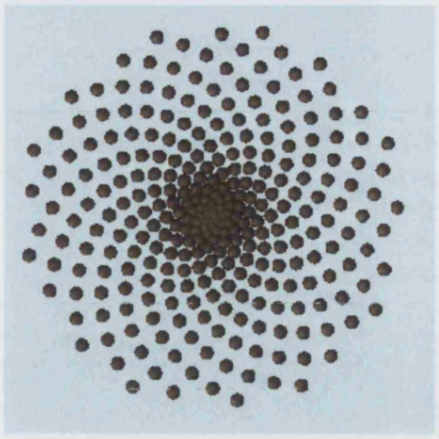


Figure 3.8: Archimedes' spiral

## 3.3 Spiral Model

### 3.3.1 Description for the models

A spiral is a plane curve traced out by a point which winds about its pole with continually increasing radius. We explain it here with a few functions. A monotonic function $\rho = F(k)$ is any function for which increases in $k$ correspond to increases in $\rho$. The increments by which $k$ is increased at each time interval must be small enough to give the effect of a smoothly flowing curve. Here are three such spirals ( where $C$ is some constant ):

*a.* Archimedes' spiral

$$\rho = C * k \tag{3.1}$$

*b.* The equiangular spiral

$$\rho = C^k \tag{3.2}$$

*c.* Fermat's spiral

$$\rho = \sqrt{C * k} \tag{3.3}$$

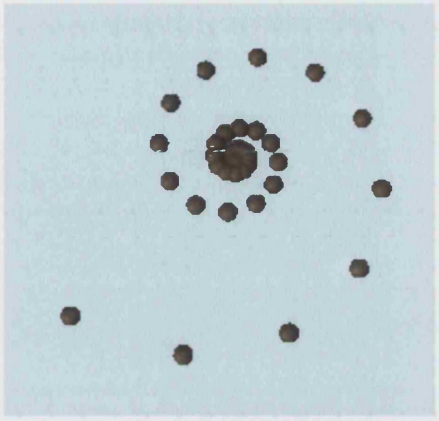


Figure 3.9: Equiangular spiral

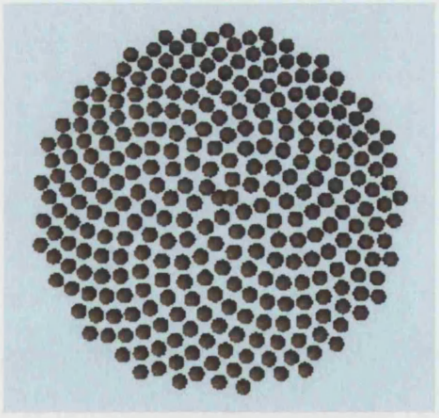


Figure 3.10: Fermat's spiral

Spiral $a$ (see Figure 3.8) increases its radius by the same amount on every turn. Spiral $b$ (see Figure 3.9) is called the equiangular spiral because its path continues to make a constant angle with the radius. It is also called the logarithmic spiral, and each turn brings about a proportional increase in radius. Sea shells and snails display this shape very clearly. Spiral $c$ (see Figure 3.10) has the property of enclosing equal areas with every turn, and we shall use this property to construct the phyllotaxis here. It is because the flower head expands at equal area with each turn, thus the Fermat's spiral formulation is suitable for the flower head development.

### 3.3.2 Implementation of spiral models

The first implementation operates in a plane and was originally proposed by Vogel [Vogel79] to describe the structure of a sunflower head. The formula is:

$$\theta = k * 137.5^{\circ} \tag{3.4}$$

$$\rho = D\sqrt{k} \tag{3.5}$$

where:

- $D$ is some constant.
- $k$ is the ordering number of a seed on the sunflower head, counting outward from the centre.
- $\theta$ is the angle between a reference direction and the position vector of the $k^{th}$ seed in a polar coordinate system originating at the centre of the capitulum.

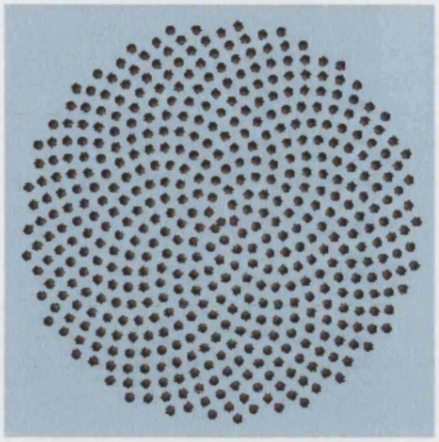


Figure 3.11: Phyllotaxis on a plane

It follows that the *divergence angle* between the position vectors of any two successive seeds is constant, 137.5°.

- $\rho$ is the distance between the centre of the capitulum and the centre of the $k^{th}$ seed, given a constant scaling parameter $D$.

The divergence angle of 137.5° is related to the Fibonacci series and the golden mean (A fact first emphasised in 1837 by the crystallographer Auguste Bravais and his brother Louis [Stewa95]). Equation (3.5) is the same as (3.3). Our implementation uses the following equation with the right-handed coordinate system:

$$x = \rho \cos\theta \tag{3.6}$$

$$y = \rho \sin\theta \tag{3.7}$$

$$z = 0 \tag{3.8}$$

In the implementation (Figure 3.11), the small sphere represents the organ of the phyllotaxis structure. From our experiments, the value of constant $D$ must

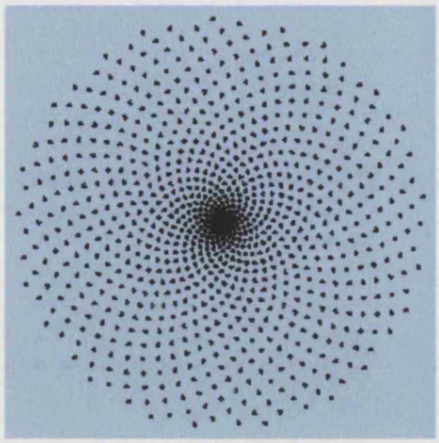


Figure 3.12: A capitulum based on Archimedes' spiral

be greater or equal to 1. And the distance between two spheres along the spiral and the gap of the spiral increases when $D$ is increased.

As a comparison, we tried using Archimedes' spiral to construct the phyllotactic pattern (Figure 3.12), that is changing $\sqrt{k}$ to $k$ in equation (3.5). For the same value of constant $D$, its distance of spheres is much greater than using Fermat's spiral. The difference of this pattern is that the sphere numbers appear twice as large as with Fermat's spiral or with reality. From the observation of a flower head, the Fermat phyllotactic pattern is a more realistic method to model a compact flower centre.

The above implementations are planar models, as all the components are on the same plane ($z$=0). However, in nature, most flower heads are not on a flat plane surface. They normally have a slightly cone-shaped centre. In the following section, we start with the basic cylindrical phyllotaxis model, then describe how to generate a cone-shaped phyllotaxis.

### 3.3.3 Cylindrical model

The spiral patterns evident in elongated organs such as pine cones, fir cones and pineapples, can be described by models that position components, in this case scales, on the surface of a cylinder [Prusi90a]. The formulas are:

$$\theta = k * \alpha \tag{3.9}$$

$$\rho = const \tag{3.10}$$

$$H = h * k \tag{3.11}$$

where:

- $k$ is the ordering number of a scale, counting from the bottom of the cylinder.
- $\theta, \rho$ and $H$ are the cylindrical coordinates of the $k^{th}$ scale.
- $\alpha$ is the divergence angle between two consecutive scales (as in the planar case, it is assumed to be constant).
- $h$ is the vertical distance between two consecutive scales (measured along the main axis of the cylinder).

Our implementation of the above model is achieved by making a choice for the $\alpha$ (we still can use 137.5°) and changing the equation (3.8) to:

$$z = H \tag{3.12}$$

Figure 3.13 shows the structure of a pineapple using spheres as organs.

Figure 3.13: A cylindrical model

### 3.3.4 The whole spiral phyllotaxis on flower head

We now consider how to model a sunflower's head using spiral phyllotaxis. The seeds of the sunflower head are not placed on a flat plane, therefore we cannot simply apply the spiral planar phyllotaxis model.

At first, we tried to generate the flower head model from the cylindrical model. The constant $\rho$ is replaced by a function corresponding to the $k$. That is, the equation (3.10) is changed to:

$$\rho_{k+1} = \rho_k - C \tag{3.13}$$

where $C$ is a small constant, the horizontal distance between two consecutive components.

However, we obtained a cone-shaped spiral phyllotaxis (Figure 3.14) which of course can be observed in some tree structures, but normally is not suitable for the flower head. It is because the flower head boundary is a curve shape, not two straight lines.

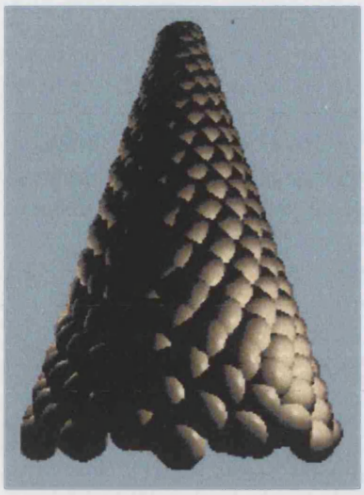


Figure 3.14: Cone-shaped spiral phyllotaxis

The simplest method for this kind of model is the combination of the planar and cylindrical spiral models. The planar model is the basis of this model. The formulas are (3.4),(3.5)and (3.11):

$$\theta = k * 137.5^{\circ} \quad (3.4)$$

$$\rho = D\sqrt{k} \quad (3.5)$$

$$H = h * k \quad (3.11)$$

Where $D$ and $h$ are constants controlling the shape of the phyllotaxis.

Figure 3.15, 3.16 and 3.17 show the sunflower head generated from the presented model. Each small sphere represents the seed on the flower head. We can see from the figure 3.16 that the model is not on a plane surface and it has a natural cone centre.

This combination model is not limited to the surface of a disk or a cylinder. In contrast, it can implement different plants by adjusting the constants and the

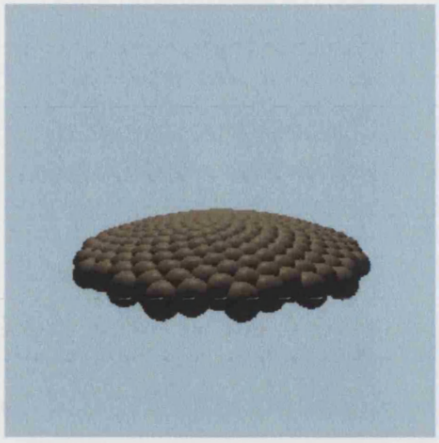


Figure 3.15: Spiral phyllotaxis on sunflower head

Figure 3.16: Spiral phyllotaxis on sunflower head, viewing from horizontal

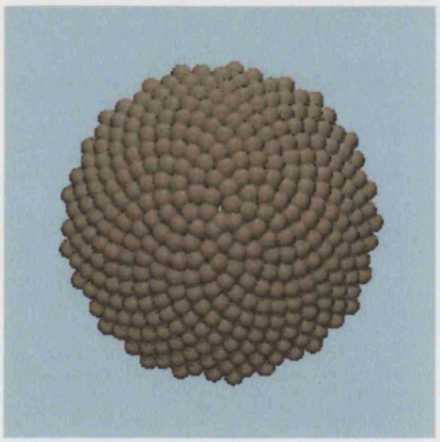


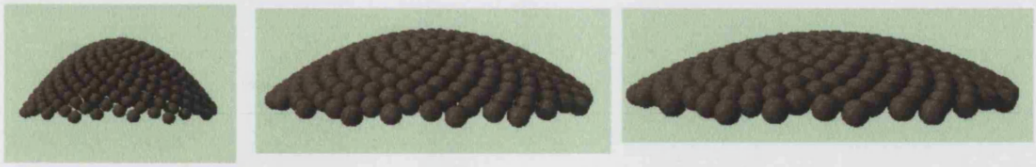
Figure 3.17: Spiral phyllotaxis on sunflower head, viewing from above

Figure 3.18: Flower: Rudbeckia Maxima

Figure 3.19: Cone flower head

characteristics of the components. For instant, we can obtain this flower head (figure 3.18) when $h$ is increased which causes the increase of the curvature in the plane, as shown in figure 3.19.

We can also simulate the growth of the flower centre by connecting the time factor with the sphere radius and phyllotaxis constants. Figure 3.20 shows the change of a centre from a small bud ($a$) to a fully open flower ($c$). The combination of these centre growth and petal change shows a realistic whole flower growth.



(a) (b) (c)

Figure 3.20: Growing flower centre

## 3.4 Phyllotaxis Applied in Flower Petals

Spiral phyllotaxis may be applied to flower petals in two main areas:

The first one is the arrangement of the petals, this includes the initial start point of the petal and the positions of all the petals. It is supposed that the whole flower head is under the spiral phyllotaxis rule. So the first petal will be put to the position after the last flower seed on the flower head. When we locate the flower petals, we also need to consider the angle and the priority of the order, which is essential for the collision detection and respond. For example, if the flower has five petals, the second petal should be located to the position when the angle with the first petal is 144°, as shown in figure 3.6, instead of 72° (which is 360° divided by 5).

The second use of phyllotaxis is for the petal collision and overlapping of petals. The phyllotaxis position of the petals will decide which petal has priority and which one will be overlapped depending on collision. Furthermore, the method automatically positions the petals according to this priority to meet the collision response.

### 3.4.1 Examples

As shown in figure 3.21, the whole flower grows with the growth of the centre and the petals. The flower centre expands in harmony with the growth of the

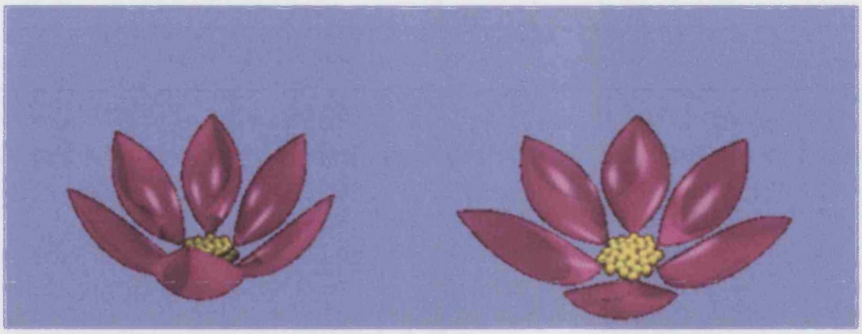


Figure 3.21: Growing flowers with centre

petals. The petal starts to be located from the last seed on the head, then growth of the seeds on the flower centre is obtained using the phyllotaxis theory. The divergence angle between consecutively formed organs (measured from the centre of the structure) is close to the Fibonacci angle of $360°\mathcal{T}^{-2} \approx 137.5°$, where $\mathcal{T} = (1+\sqrt{5})/2$. However, the angle between the consecutively formed petals is 144° which is explained in the last section. There is no collision for the petals in this example, this phenomena will be shown after we describe the collision detection and respond in the later chapters.

## 3.5 Conclusions

Of course, nobody is suggesting that botany is quite as perfectly mathematical as the phyllotaxis model presented here. In particular, in many plants the rate of appearance of primordia can vary considerably. In fact, changes in morphology (whether a given primordium becomes a leaf or a petal) often accompany such variations. So maybe what the genes do is affect the timing of the appearance of the primordia. But plant do not need their genes to tell them how to space their primordia: that is done by the dynamics. However, by using a mathematical

model to summarise the biological rule, it provides us with a possible method to build a natural virtual environment.

In this chapter, we have presented some important biological background materials for the whole model. With the comprehensive reviews of the phyllotaxis and Fibonacci theory for plant organs, we built the spiral phyllotaxis model for a flower head. This gave a simple explanation of how the positions are achieved for the seed on the flower centre and also the positioning of petals.

We concluded the chapter with illustrations of the growth development for the flower centre and petals using spiral phyllotaxis.

# Chapter 4

# Surface Modelling for the Flower Petal

## 4.1 Introduction

In this chapter, we propose a method for generating flower growth animation in which petal surface and shape can be changed simultaneously in real time. We represent a flower petal as a set of control points defining a bicubic patch.

In the real world, changes of shape are very common. Although you might think that a change of scale also implies a change of shape, a change of scale does not constitute a change of shape in the sense meant here. Scaling an object makes it uniformly longer or shorter in one or more directions, but it does not alter the configurations of the surface. In order to achieve any kind of successful animation, we have to be able to animate changes in an object's shape. Animating a shape change involves animating the positions of the points that define the surface of the object. So it is very important to choose a surface representation which is suitable for the animated object and with easy access to control points.

It is obvious that petals or leaves change in shape and form during their life cycle, it is important to construct a suitable dynamic surface model to represent this life-cycle. Some such surface models have been developed over the past decade. Guo describes a method for reconstructing an unknown surface [Guo97a] from a set of scattered points. Welch presents a method [Welch92a] for the interactive modelling of free-form surfaces, where the user is free to manipulate the control points to obtain different shapes for the same surface with constraints. Durikovic presents the shape of the organ by a number of ellipsoidal clusters centred at points on the skeleton [Durik98a]. He also introduces several tables with which to store the database of statistical geometry of organs, such as size, growth speed, among others. However, these surface representations with predefined final shape or data sets are not suitable for our petal surfaces. In our system, the final shape of the petal surface is not predetermined, and in addition, the surface can be constructed and controlled with the growth functions and user's interventions. In order to obtain this flexible and controllable surface, we will use a bicubic patch with sixteen control points for the petal surface. We start with a basic description of a flower petal shape, then describe the petal surface model in detail, along with the controlling method and important model features.

## 4.2 Flower Petal Shapes

### 4.2.1 Flower structure

Most flowers contain two sets of sterile appendages [Holm79], the sepals and petals, which are attached to the receptacle below the fertile parts of the flower, the stamens and carpels (see figure 4.1). The sepals occur below the petals,

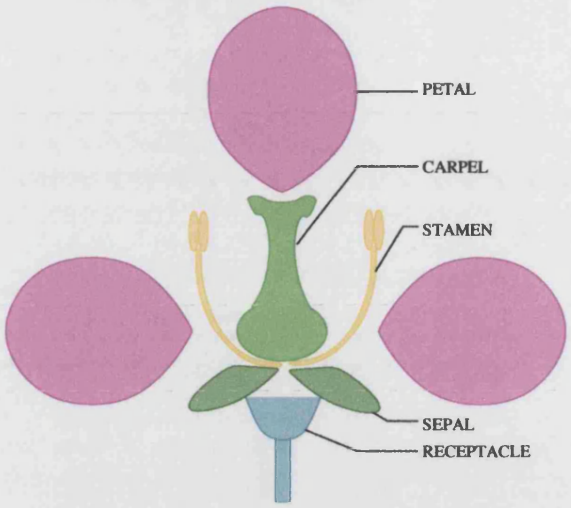



Figure 4.1: Flower structure

and the stamens below the carpels. The sepals and petals are essentially leaflike in structure. Commonly, the sepals are green and the petals brightly coloured, although in many flowers both parts are similar in colour. The purpose of petals is to attract insects or animals. And in some cases, plants have evolved highly specialised flowers to attract only one species of insect or animal. Petals therefore vary enormously, and they tend to be more fully developed in flowers pollinated by animals.

A typical petal life-span is usually just sufficient until the flower has been pollinated. Then, they wither, usually quite quickly, and fall. But if pollination does not take place, the petals may remain on the flower for some time. Orchids have both the longest and shortest living flowers: nine months (Grammatophyllum multiflorum) and five minutes (Dendrobium appendiculatum, in which all the flowers open simultaneously and mass pollination occurs).

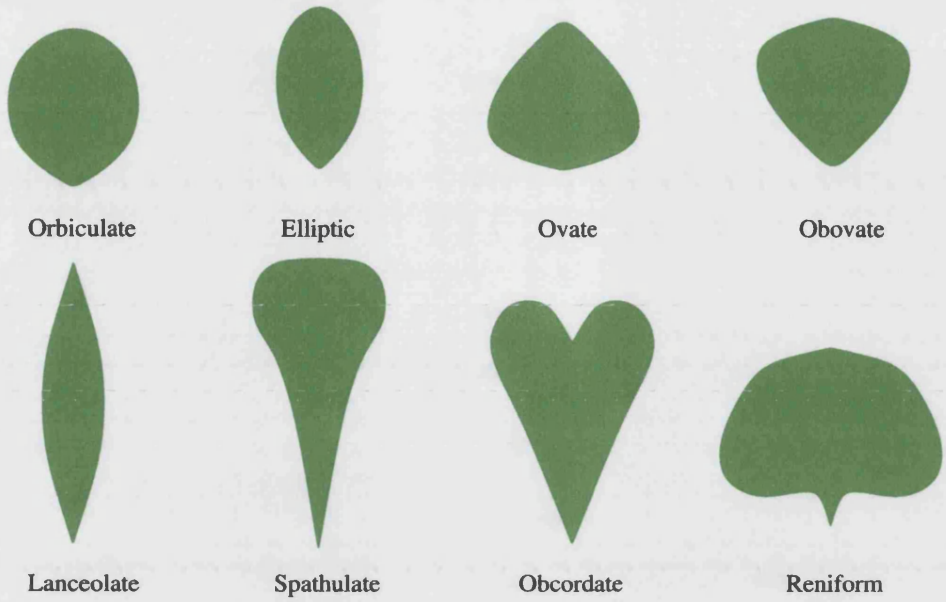



Figure 4.2: Base leaf shapes

### 4.2.2 Petal shapes

There are two major petal shapes:

- Leaflike shape petal

  The base leaf shapes [Benso79] are shown in the figure 4.2:

  Most of these shapes are easy to model with one bicubic patch. The difficulty is to show the surface changing in detail especially the ruffled edge.

- Evolutionary shape

  This mainly includes non-divided or part-divided petals and pollinated shape petals (shape change involved structural features developed to exclude some pollinators). For example, long strap-shaped petals in species such as the sunflower, daisy and black-eyed Susan. In addition this also includes the cuplike lip, a petal modified into a landing platform for insects, such as in the orchids 4.3.

Figure 4.3: Orchid photo

In our system, we focus on a leaflike petal shape. In principle, our system could model evolutionary shapes, though this is beyond the scope of the work presented here. However, same issues involved in modelling evolutionary shapes are discussed in the conclusion.

## 4.3 Surface Representation

To achieve a realistic result for the development of a bud into a flower, it is inevitable that the petal model must possess aspects of the complexity of a real petal. The model needs to provide satisfactory surface continuity and smoothness as the flower develops. This presents problems for the construction of the 3D model for our simulation. Building a model of this complexity using traditional polygonalization techniques would involve prohibitive storage and processing overheads, therefore alternative methods must be used.

### 4.3.1 Bicubic surface patches

Prusinkiewicz and Lindenmayer showed how plant components, such as stamens, petals, leaves, seeds, can be built out of bicubic patches [Prusi90a]. A patch is defined by three polynomials with degree three, with respect to parameters $s$ and

$t$. The following equation defines the $x$ coordinate of a point on the patch:

$$\begin{aligned} x(s,t) &= a_{11}s^3t^3 + a_{12}s^3t^2 + a_{13}s^3t + a_{14}s^3 + \\ & a_{21}s^2t^3 + a_{22}s^2t^2 + a_{23}s^2t + a_{24}s^2 + \\ & a_{31}st^3 + a_{32}st^2 + a_{33}st + a_{34}s + \\ & a_{41}t^3 + a_{42}t^2 + a_{43}t + a_{44} \end{aligned}$$

Analogous equations define $y(s,t)$ and $z(s,t)$. All coefficients are determined by interactively designing the desired shape. Complex surfaces are composed of several patches.

### 4.3.2 Bézier patch

Just as with two dimensional curves, three dimensional patches may use a variety of control strategies, including Bézier, Hermite, and B-Spline bases. As described in the previous chapter, a Bézier patch requires sixteen control points. The four corner points control the position of the patch and lie on its surface. The intermediate twelve points control the tangents to the patch along the edges and at each corner and may be used by the designer to "pull" the surface of the patch into the shape desired (figure 2.2) [Farin88a, Barte87a].

The Bézier form of the bicubic parametric patch has a very concise matrix formulation specifying a vector point, $P(s,t)$, on the surface in terms of the sixteen control points. This relationship is expressed as [Fireb93a]:

$$P(s,t) = SBPB^TT^T \tag{4.1}$$

where:

$$S = \begin{bmatrix} s^3 & s^2 & s & 1 \end{bmatrix} \text{ ($s$ parameter row vector)}$$

$$B = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \text{ (Bézier coefficient matrix)}$$

$$P = \begin{bmatrix} P_{00} & P_{01} & P_{02} & P_{03} \\ P_{10} & P_{11} & P_{12} & P_{13} \\ P_{20} & P_{21} & P_{22} & P_{23} \\ P_{30} & P_{31} & P_{32} & P_{33} \end{bmatrix} \text{ (Control point matrix)}$$

$B^T$ = Transposed $B$ matrix (switch rows and columns)

$$T^T = \begin{bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{bmatrix} \text{ (Transposed $t$ parameter vector)}$$

For a regular and symmetrical petal, we can keep $P_{00}, P_{01}, P_{02}, P_{03}$ as one control point, thus making the control easier. To make it simpler, $P_{30}, P_{31}, P_{32}, P_{33}$ could be combined as one point if the top of the petal is a discontinuity point. Then it forms the lanceolate shape in figure 4.2. However, spreading the control points enables more controls for more complex surface shapes. As shown in figure 4.4, four control points are located at the bottom of the petal while the base portion of the petal is simple. The other two groups of four control points usually control the middle part of the petal. The last four control points play an important role in shaping the upper part of the petal when they are located at the top of the petal. When we consider the structure for the petal surface, we need to pay attention to where the discontinuity points are, in which the two groups of the four control points can be joined, forming a corner control point. The part

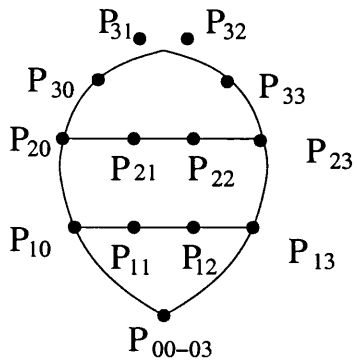



Figure 4.4: Petal structure with sixteen control points

of the surface formed by the same group of four control points will generate a smooth and continuous curve. So the surface structure will be changed if the discontinuity points are different. We will show this effect in the next section.

### 4.3.3 Advantages

Bicubic Bézier patches have become a popular tool for surface modelling. The obvious advantages include:

- *Ease of interactivity* – the control point effects are readily observed and understood, and the control points themselves are easily modified, either numerically or interactively;
- *Representational efficiency* – complex surfaces are represented by a very small set of numbers.

So this approach is applied here as our preferred method for surface representation for our flower petal modelling.

Figure 4.5: Symmetrical petal: (a) four control points are at the top; (b) the top four control points generate the curve.

## 4.4 Petal Shape Examples

The bicubic patch allows us to model a wide range of petal shapes. It is therefore worthwhile to consider in more detail the shapes we can obtain by this method.

From graphics theory, we know that one group of four control points will form a curve (see figure 2.1). As shown in figure 4.5 with two symmetrical petals, one group of four control points give the representation of the bottom of the petal. In the left petal ($a$), four control points are on the top of the petal, superposed on each other, therefore representing only one position. So the petal will very much depend on the remaining two groups of eight control points to pull it into the desired shape. The figure 4.6 shows all control points from the back of the petal. In the right petal ($b$), the top four control points generate a curve, which will control the top part of the petal. So, the three groups of control points will determine the petal shape. From these two figures, it is obvious that the bicubic patch is effective in shape control for the petals.

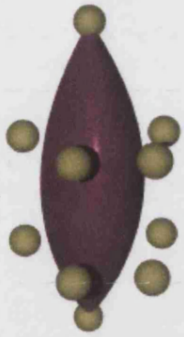


Figure 4.6: Symmetrical petal: four control points are at the top, four control points are on the bottom.

Figure 4.7: Symmetrical petal: (a) the top four control points pull down the curve; (b) the middle two control points on the top pull further down.

### 4.4.1 Controlling shape change

Figures 4.7 and 4.8 show how the control points control the shape of a petal. From the same initial bicubic structure, the petal surfaces can be varied from the simplest shape shown in figure 4.5(a) to a more complex shape shown in figure 4.8(a). This is simply achieved by pulling two control points downwards or inwards. Naturally, an asymmetric shape will be obtained when the control points are not located symmetrically, as shown in figure 4.8(b).

Figure 4.8: (a) Symmetrical petal: two control points on the top end pull inward; (b) Asymmetrical petal: the position of two top control points are exchanged.

### 4.4.2 Discontinuities and control points

From the theory of bicubic patches, we know that a group of four control points will generate a continuous curve. Thus, if we want a discontinuity point, we will have to avoid placing any of the middle control points at the desired discontinuity. Instead we must place the corner control points at the discontinuity point. For example, if we want to generate the heart-shaped petal (see figure 4.9), there are two approaches we can consider when we use one bicubic patch to represent this shape. The first approach is to place a group of four control points together at the top as the discontinuity point. It is obvious that this will reduce the number of control points, which in turn leads to less control for the other part of the surface. The second way (shown in figure 4.10) is to use two groups of four control points on the left and right site of the petal separately and they are joined on the middle top and bottom of the petal as corner control points, just like a diamond shape. This has the advantage of retaining all the control points, but has implication for the growth of the petal. Thus it requires a slightly different growth rule for the surface as we need to make sure the relevant positions of the control points maintain a heart-shaped petal.

Figure 4.9: Symmetrical petal: discontinuity on the top.

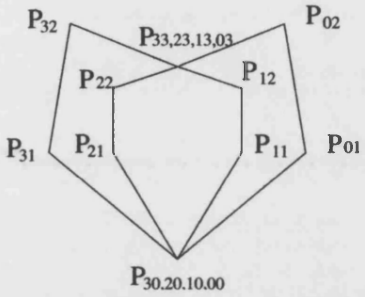



Figure 4.10: Diamond shape arrangement for petal control points

## 4.5 Description of Petal Surface Change

### 4.5.1 Factors controlling growth

With the theory for the growth rate, we need to consider what parameters are necessary for the petal shape control. However, to enable a non-expert user, with only a general understanding of the petal shape, to effectively control the petal growth we have restricted the control factors to three. These factors are the length, width and curvature growth factors. Increasing the parameter for the length factor will generate a longer petal for a given time interval. In a similar way, increasing the width factor will generate a wider petal. Finally increased curvature factor will accelerate the petal surface curvature growth. These three control factors are sufficient to achieve a wide variation of the growth rate and surface definition of flower petals.

For a regular and symmetrical petal, we can locate the sixteen control points evenly around the petal. The growth between two groups of four control points will be considered as having the same growth rate. For example, if the top four control points grow 1 unit value for the time interval, the next group of four control points grows $\frac{2}{3}$ unit value. The third group grows $\frac{1}{3}$ unit value if the bottom group control points remain on the petal base.

In summary, the main three growth factors play an important role in changing the initial petal structure to the desired final form. For example, the three factors allow the user to control a bud growing into a flower with petals of different length, width and curvature at all stage of growth.

### 4.5.2 Implementation

In the early stages of this research, the implementation demands a large amount of artificial control. We need to set up the growth equation by setting the values of the growth rate. The growth rate value in length, width and depth and the growth factors for all the control points must be input to calculate the new positions after the specified time interval. The selections of all the values in these two steps depend upon the desired petal shape. The growth factors for some control points need to be reset if their desired surface growth tendency is changed. Then the new control points matrix will be generated, and thus the patch surface is formed. The coordination of moving the control points are controlled by the user. We will refine this user-intensive method and apply an improved model in the later stages of this research, as reported in subsequent chapters.

Figure 4.11: Petal growth: length rate:0.8, width rate:0.6, depth rate:0.2.

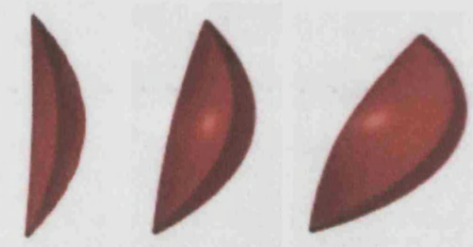


Figure 4.12: Petal growth: length rate:0.0, width rate:1.0, depth rate:0.1.

### 4.5.3 Growth examples

Figure 4.11 and 4.12 show examples of the growth processes. Generally, the petal surfaces are getting longer, wider and flatter, thereby opening away from the flower centre with increasing time. From these figures, we can see there are three major advantages in using our petal surface model. Firstly, varied surface shapes can be obtained easily by changing the growth rate value in length, width and depth direction or changing the growth function. Consider the flowers shown in figures 4.11 and 4.12. In figure 4.12 the flower has wider petals with a greater width growth rate. Secondly, the growth tendency can be interrupted by changing the growth rate for a particular factor at any time. Again, comparing figure 4.11 and 4.12, which both have the same growth rate at the beginning of their cycles, we see that figure 4.12 terminates with a different shape as a result of increasing

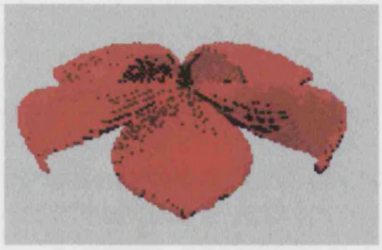


Figure 4.13: Flower petals at a mature stage

Figure 4.14: Flower petals development after the mature stage

the width growth rate. Thirdly, asymmetrical petals can be obtained by changing the position factors for the control points, that is, moving the relative positions between all the control points.

Furthermore, the easy user access to the control points and surface shapes provides the possibility of generating the petals from any intermediate stage, as shown in figures 4.13 and 4.14, with hanging and wrinkled petals. In summary, it provides a direct method of surface control. Compared to using predefined surfaces or generating surfaces from fixed original and final shapes, this method is more flexible and allows th user to exercise considerable creativity in surface development.

### 4.5.4 Convex hull

As we mentioned previously, we must also consider collisions between petal surfaces when we are animating growth. Here we discuss an important feature of the petal surface for this task. A Bézier surface has an attractive convex-hull property which is very useful for our surface intersection test [Farin88a]. Each of the boundary curves of a Bézier surface is a Bézier curve. Considering the defining polygon net for the bicubic Bézier surface shown in Figure 2.2, it is easy to see that the tangent vectors at the patch corners are controlled both in direction and magnitude by the position of adjacent points along the edges of the net. Consequently, the user can control the shape of the surface patch without an intimate knowledge of tangent or twist vectors (They are the cross-derivatives at the each end-point, specify the rate of change of the tangent vectors with respect to $u$ and $v$ (ref chapter 2). They are vectors normal to the plane containing the tangent vectors).

A Bézier curve is determined by a defining polygon. The curve generally follows the shape of the defining polygon, and the first and last points on the curve are coincident with the first and last points of the defining polygon. In addition, the curve is contained with the convex hull of the defining polygon. These properties mean that the curve is within the largest convex polygon obtainable with the defining polygon vertices. In Figure 4.15, the convex hull is shown by the shaded area. The convex hull for 2D curves is the convex polygon formed by the four control points. For 3D curves, the convex hull is the convex polyhedron formed by the control points.

Analysing the petal growth process from a theoretical viewpoint, we can see the control points moving through space and thereby changing the shape for the

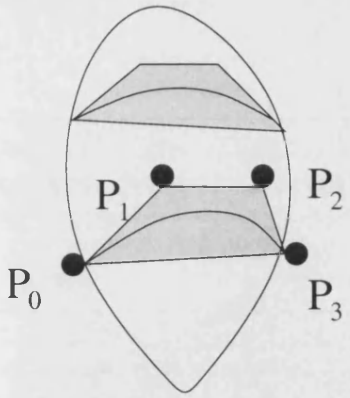



Figure 4.15: Bézier curves and convex hulls with defining control points.

petal by the intuitive definition of a surface. We will now formalise this intuitive concept in order to arrive at a mathematical description of a surface. First, we assume that the moving curve is a Bézier curve. At any time, the moving curve is then determined by a set of control points. Each original control point will move through space according to the growth function. Our next fact is that this curve is contained with the convex hull.

In Figure 4.16, small spheres represent the control points for the petal patch surface. In the first flower, the control points of the adjacent petals are crossed over, that is the control points are within the boundary of the adjacent petal, not within their own petal. The control points will be moving apart as the flower grows and the petal surfaces separate. In Figure 4.17 , the control points of the adjacent petals in the upper rectangle are moving apart, which implies that the upper part of the petals do not collide. However, in the lower rectangle the control points have crossed each other and are therefore out of order, indicating that the lower part of the petals may be intersecting. The four control points in the vertical boundary of the petal surface also form a convex polyhedron. Therefore the collision detection between the two adjacent petal surfaces can be

Figure 4.16: Petals with sixteen control points

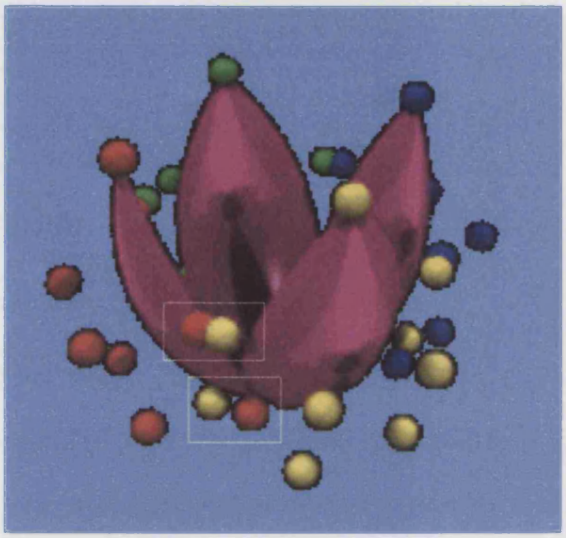


Figure 4.17: Control points crossed over

converted to two horizontal convex polyhedron tests (from the two middle sets of control points) and one vertical convex polyhedron intersection test.

If there is no intersection of these polyhedron, then there is no collision between these two surfaces. If there is an intersection of the polyhedron, then we repeat the test on the subdivision convex polyhedron which is closest to the side of the adjacent petal surface and recursively repeat this test. So this feature will be very useful in cutting the collision computation cost which is discussed in the later chapters.

## 4.6 Conclusions

In this chapter, we presented the theory of flower shape, the purpose of which is to give a good description for the shape of flower components. We have also proposed a method for generating flower growth animation in which the petal surface and shape can be changed simultaneously in real time. In addition, we presented the flower petal shape and petal surface structure theory.

We introduced a smooth surface model with a bicubic patch for the petal, which can simulate the surface development with easy access to individual control points. At the same time, the factors in length, width and depth represent the principle growth for the petal.

This chapter also presented a description of various kinds of surface growth control, step by step with analysis of the petal shape changes, and illustrated the use and results of some growth factors in some common cases. They may be assumed to be fixed relative to the growth surface or to the body being generated.

We concluded the chapter with a discussion of the convex hull feature which will be useful for surface collision detection. In summary, we believe that the proposed surface modelling method and its extensions will prove useful in many applications of plant modelling, from research in plant development and ecology to the surface design of plant organs and in the production of animated plant models for use in virtual environments. We will discuss further how the flower grows with a natural growth function in the next chapter.

# Chapter 5

# Flower Growth Functions

## 5.1 Introduction

In this chapter, we present the theory of growth functions, the purpose of which is to give a good description of plant development. In addition, we expand growth function theory to enable the growth rate to vary as the petal develops. We also develop a graphical representation for plant growth functions and introduce the integral equations for growth measurement based on the specific time interval. And we also describe the shape control given by the growth function.

We may ask the question: Why do we need a growth function?

Measurement, analysis, and visualisation of plant growth is of primary interest to plant biologists. Loomis [Loomi97] developed some software tools to support such investigation. Two forms of growth visualisations for plant roots and plant stems are presented in his work. Root growth is described by the image-space method, which is an adaptation of a parametric deformation method for image matching. In this model, a rubber-sheet mapping between two images is formulated to contain the shape change information for the growth. Their plant model

is a tubular object formed by a 3D spine with a thickness that can vary along the length, and provides some parameters to match growth features from the image. The growth model and function cannot be applied in the general case or other biological models. More complex deformation (such as development of leaves) is suggested as future work by the author.

Simulating natural tree features, such as leaves, branching, flowering, etc, that are acquired in a growth process still remains a challenging problem. A growth model having the abilities of growth-regulations for simulating the visual nature of botanical trees was developed by Chiba [Chiba94a]. Since an irregular botanical shape is always generated during a growth process by the growth environment, such as sunlight conditions and random pruning of branches (caused by storms or gardeners), not only do trees not have a regular shape, but no two trees are identical, even if they are the same species. In their research, they present an improved growth model while taking into account growth constraints, thus showing that the tree shape is significantly influenced by its growth environment. The presented growth functions demonstrate the effects of the growth constraints and its importance for the simulation. However, the work is only related to the shapes of tree skeletons, and the growth model is not suitable for the shapes of leaves, blossoms and fruits.

In the past, few researchers have focused on the plant organs, such as leaves or flowers. However, these organs are essential for giving natural and seasonal variation. In addition, these organs are particularly important when simulating a plant at close range. In the further work of Chiba [Chiba96b], he presented simulation methods for leaf arrangement and colour evolution based mainly on the estimation of the amount of sunlight and the brightest direction of each part of each leaf. The growth model described above is used to demonstrate the

representation methods for leaves. However, this model has no ability to simulate the leaves' growth process. So we have to find another method for our growth simulation, which will enable us to describe a realistic life-cycle for the evolution of a plant form..

Growth is a continuous process, but simulation models operate in discrete time steps, making it convenient to simulate the discrete addition of structural units. Consequently, the structural unit considered by a model, the time taken for a unit to appear, and the time step used by the model are usually related. A close approximation to continuous growth can be achieved by solving differential equations of very small time steps [Prusi93b], which solved problems associated with the growth of trees. In addition L-systems can be used to simulate continuous growth of plant organisms. Lindenmayer [Linde76] introduced parallel rewrite systems to describe the growth of living organisms. The theory related to these systems, called L-systems, developed very quickly. As L-systems are recursive solutions of sets of rewrite rules, we cannot offer a continuous solution. The end stage in a plant's development must be defined by restarting the recursion, for example, a six recursion must be developed from the previous five stage recursions. Because of the deficiencies of L-systems in modelling growth, we have chosen the differential equations as the method for solving all intermediate stages of flower growth.

We start with the description of the general growth function. After criticising the earlier growth function used in our previous plant development, we present our growth function along with same graphical illustrations. Then we conclude the chapter with an explanation of the controlling growth factors.

## 5.2 General Growth Function

We wish to find a growth function which will give a good description of plant growth data, its life-cycle, over a whole season. If possible, the parameters of the growth equation should be physiologically meaningful, relating either to the environment or to the plant. Such an equation may be used to summarise data, and possibly to interpret growth data from experiments using plants of different varieties or with different environmental conditions.

If we think of an organism as being composed of a population of cells of fixed size, then the density-dependent model of population growth will describe an organism. This theory states that, as the number of cells increases, the amount of resources for cell division decreases, reducing organism growth rate. The differential equation for this is the familiar logistic equation. We need to make some assumptions when we solve the above problem [Thorn76a]:

1. The plant is completely defined by its dry weight $W$. Thus the system is described by a single state variable $W$, where $W$ is a dependent variable, and varies with time $t$, where $t$ is an independent variable.

2. Growth occurs at the expense of a single substrate $S$.

3. The rate of the growth reaction is linearly proportional to the substrate level $S$, and also to the plant dry weight $W$, so that the growth is autocatalytic (catalyses its own production or activates a substance that produces it). The rate of the growth reaction is $kWS$ where $k$ is a constant.

As a consequence of the above assumption (3), it follows that

$$\frac{dW}{dt} = kSW \tag{5.1}$$

This first-order differential equation cannot yet be solved, because the substrate level $S$ will vary as growth proceeds. If $W$ and $S$ are measured in the same units, and there is no loss of material when converting $S$ into $W$ by the growth reaction, then we can say

$$dW = -dS; \tag{5.2}$$

this equation states that an increment in dry weight is exactly matched by a loss in substrate. We also have the following equation:

$$W + S = W_i + S_i = constant, \tag{5.3}$$

where $W_i$ and $S_i$ are the values of $W$ and $S$ at time $t = 0$, and denote the initial conditions. Equations 5.2 or 5.3 simply express the conservation of matter. Since $W$ and $S$ are not allowed to be negative (such values would be physiologically meaningless), it is clear from equation 5.3 that $W$ will have its maximum and final value when there is no substrate left, and $S = 0$. Equation 5.3 may be re-written as

$$W + S = W_f = W_i + S_i, \tag{5.4}$$

where $W_f$ is the maximum value of $W$. Substituting for $S$ from equation 5.4, equation 5.1 becomes:

$$\frac{dW}{dt} = k(W_f - W)W. \tag{5.5}$$

This equation is a statement of the model in differential form. As the system has only one state variable, only one equation is needed. Dynamic models (that is, those in which time is an independent variable) are usually stated in

terms of one or more first-order differential equations. Generally, these equations cannot be solved analytically (that is, an algebraic solution cannot be found), and it is necessary to integrate the equations numerically to obtain numerical solutions. We will present what we need from this mathematical model for our plant development in the later sections.

## 5.3 Plant Growth Function

As a similar case, continuous processes such as plant growth need to be described by growth functions. A popular example of the growth function is a sigmoidal type, monotonically increasing from minimum to maximum with growth rates of zero at both ends of time interval $(To, T)$, as shown in figure 5.1. The growth is slow initially, accelerating to reach a maximum, then slowing again and eventually ceasing. We apply the popular Velhurst's logistic function, defined by the equation [Edels88]:

$$\frac{dz}{dt} = l * z(1 - \frac{z}{z_{max}}) \tag{5.6}$$

($t$: time; $l$: length growth rate; $z$: the growth in $z$ axis direction)

Usually, bicubic parametric patch modelling suffers from lack of a high-level modelling abstraction for shape control. So the growth function can be applied in the $x, y, z$ directions for each control point. It is presented in equation 5.6 for growth in the $z$ axis direction.

The time interval is chosen by the user, according to the smoothness of the desired animation sequence. This time interval is then fixed during the recursive

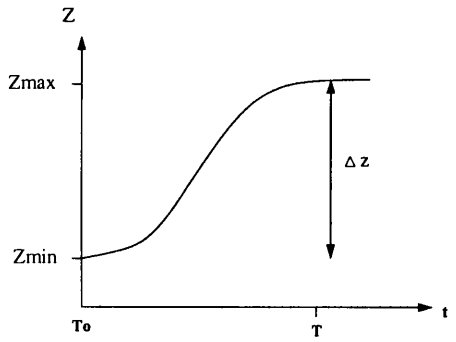



Figure 5.1: General growth function

process to determine the surface features. The flexible growth rate curve allows variation for individual control points. It means that we can apply a different growth function for one control point or alternatively to a group of control points. A unique irregular petal will be generated when one or some control points have different growth tendencies. In our improved model, the growth function will be applied to the control points by the relevant force, which we describe at a latter stage.

In the paper by Prusinkiewicz [Prusi93b], a cubic function of time was chosen as a growth function. As presented:

$$x(t) = -2\frac{\Delta x}{T^3} * t^3 + 3\frac{\Delta x}{T^2} * t^2 + x_{min} \tag{5.7}$$

where $\Delta x = x_{max} - x_{min}$ and $t \in [0, T]$. The equivalent differential equation is:

$$\frac{dx}{dt} = -6\frac{\Delta x}{T^3} * t^2 + \frac{\Delta x}{T^2} * t = 6\frac{\Delta x}{T^2}(1 - \frac{t}{T}) * t \tag{5.8}$$

with the initial condition $x_0 = x_{min}$. In order to extend this curve to infinity, it is also defined:

$$\frac{dx}{dt} = 0 \tag{5.9}$$

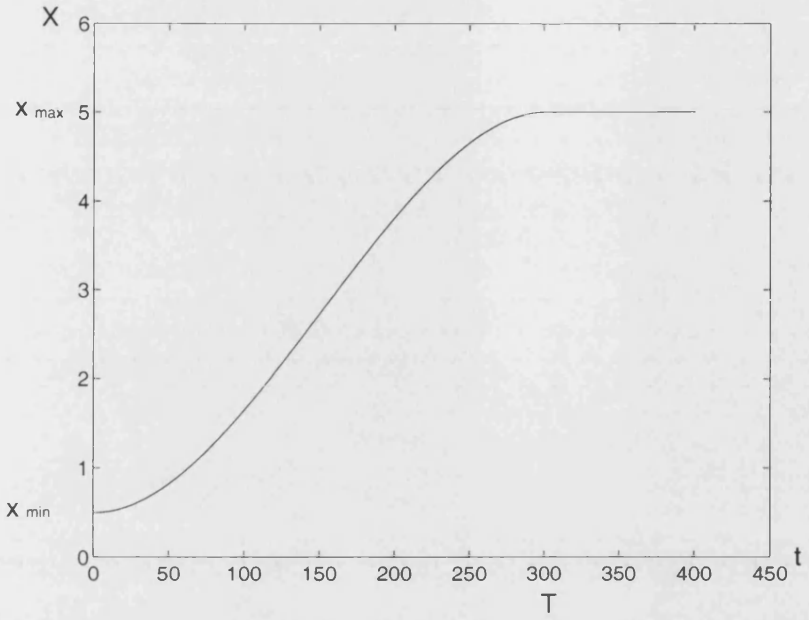



Figure 5.2: Growth function from paper [Prusi93b]

for $t \in (T, +\infty)$.

Equation 5.7 is shown by the figure 5.2. It produces $x_{max}$ when $t = T$. The shape of the curve and the change rate of the growth rate depend on the value of $\Delta x$. When $\Delta x$ increases, the curve steepens, resulting in greater growth for the same time interval $T$. It is obvious that the curve is entirely defined by the value of $\Delta x$, therefore the desired perturbation during the growth period is impossible. Thus for a chosen $\Delta x$ we will always obtain a given life-cycle. However, in natural growth, different environments cause different growth curves for the whole growth procedure even though the plant arrives at the same final growth value, as shown in figure 5.3. This figure more accurately describes what we would find in nature.

From [Prusi93b], equation 5.8 and 5.9 are represented by the figure 5.4. These equations define a zero growth rate at both ends of an interval $T$, and within time interval $T$ its value increase from $x_{min}$ to $x_{max}$. However, evaluation of these equations, as illustrated in figure 5.4 shows that the growth change rate

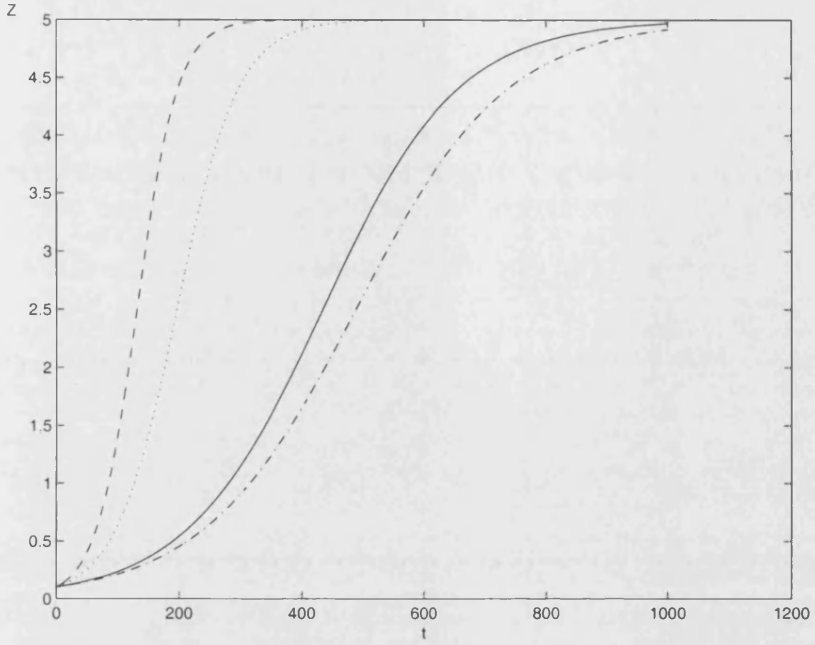



Figure 5.3: Growth function with different growth rates

(that is, the second derivative) is maximum at the beginning and the final stage of the growth process. In addition, it has a discontinuity point at $T$. This is obviously completely unrealistic when compared against the biological principle in which the plant growth initially increases in value slowly, then accelerates, and eventually nears the maximum value or arrives at the level for this growth season. The end rate of change of the growth rate (the second derivative of the growth function) should slow down and not reach a maximum as shown in figure 5.4.

A more serious problem will become apparent if the plant continues growing, for example to display seasonal growth or daily growth. In figure 5.5, we can see that the growth change rate function has a discontinuity point between each growth cycle, that is if we suppose the plant keeps growing after one growth cycle. However, it is very common for the plant to start growing again after one night or one season, so again the formulation chosen by [Prusi93b] leads to completely

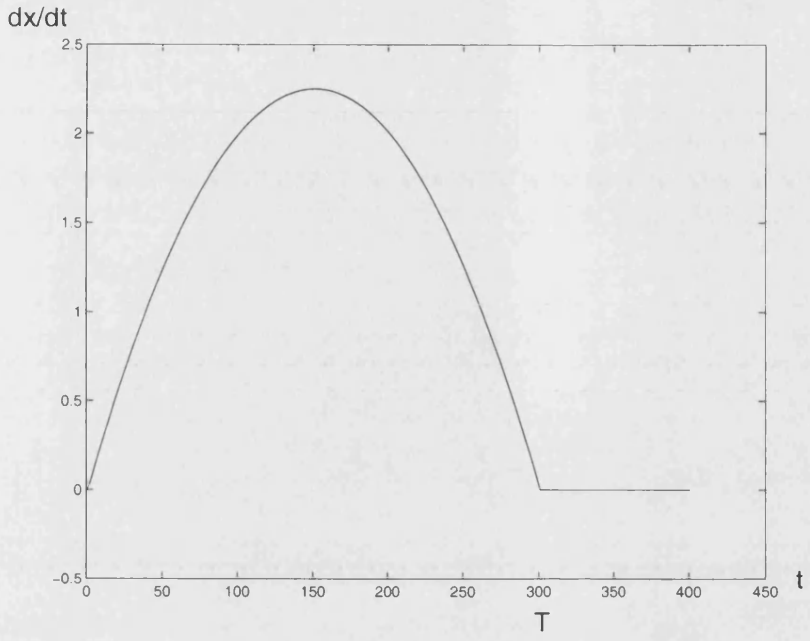



Figure 5.4: Growth change rate in the paper [Prusi93b]

unrealistic results. So we have to find a proper growth function which presents the growth cycle correctly in terms of observed natural growth. All details are discussed in the next section.

## 5.4 Growth Function with Perturbations

In reality, the whole biological growth process includes a number of growth cycles, which vary from minutes to months. It also involves different kinds of perturbations, such as climate effects, organic effects, and artificial perturbation caused by human intervention. In previous work [Prusi93b], the growth of a biological organism is determined by equations, which a-priori fixed the final form of the organism even before the growth starts. Here, we reject this approach and instead seek methodologies that can enable the growth function to take account of natural perturbations.

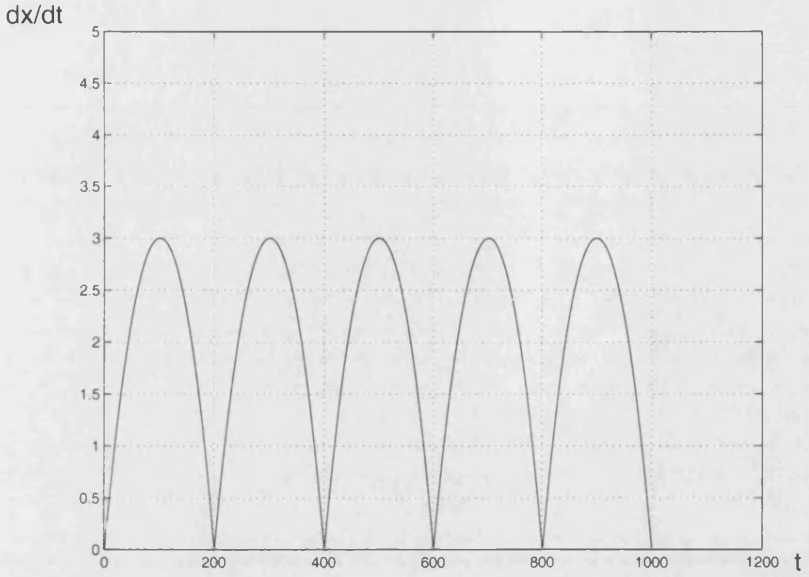



Figure 5.5: Continue growth will cause discontinuity point between two cycles

When we consider a growth function which consists of several growth cycles, we need to take account of how the growth cycles are connected, that is, how the plant grows from the last growth cycle, and the relationships between different growth cycles. Normally, the general growth function is only applied for one complete growth cycle, and cannot be used for any growth process with previous growth. As shown in figure 5.6, curve $c$ is the general growth function with a very small minimum start value (near zero). The curve shows that the growth is slow initially, accelerating near the maximum value stage, slowing again and eventually ceasing. However, the curves $b$ and $a$ do not show this feature when the growth has already started and with large (greater than 1) minimum start value. So it is obvious that it will have a discontinuity if we apply this growth function with continuing growth from the previous growth cycle.

Continuing growth is quite common in the natural world. If we take the example of flowers, the growth of the petal slows down or even ceases during the

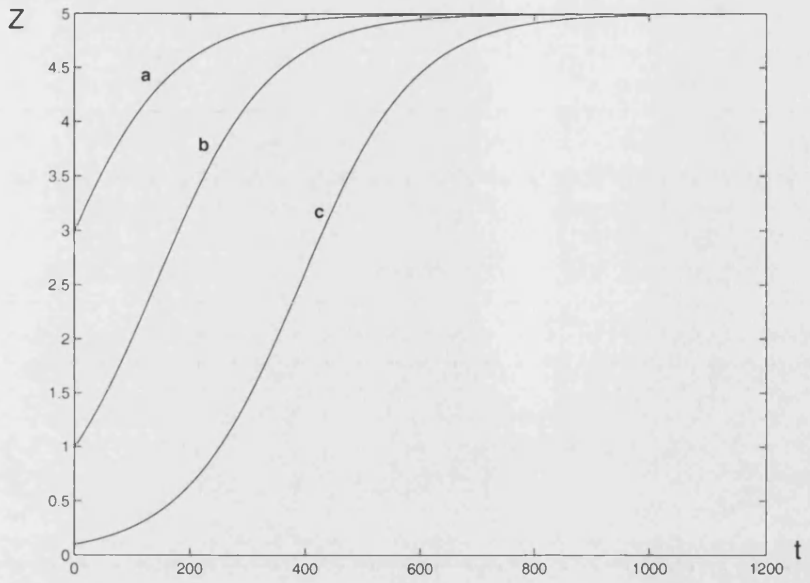



Figure 5.6: Growth functions with different start values

night and then returns to growing again the next day. So a growth function with the feature of continuity from a growth cycle to the next cycle will be an important part of our research and implementation. Since the growth continues from a previous growth cycle, the concepts for minimum growth length and maximum growth length are changing, that is, the maximum growth length for a cycle will be the minimum length for the next growth cycle, and our growth function should therefore be changed to:

$$\frac{dz}{dt} = l * \Delta z * (1 - \frac{\Delta z}{\Delta z_{max}}) \tag{5.10}$$

where $\Delta z = z - z_{min}$ and $\Delta z_{max} = z_{max} - z_{min}$.

As shown in figure 5.7, curves $a$, $b$ and $c$ derived from 5.10 now all display correct growth feature, which has slow initial growth rate then starts to accelerate. Compared to the growth function shown in 5.6, this function will provide a smooth connection with the previous growth value for each curve 5.9. Figure

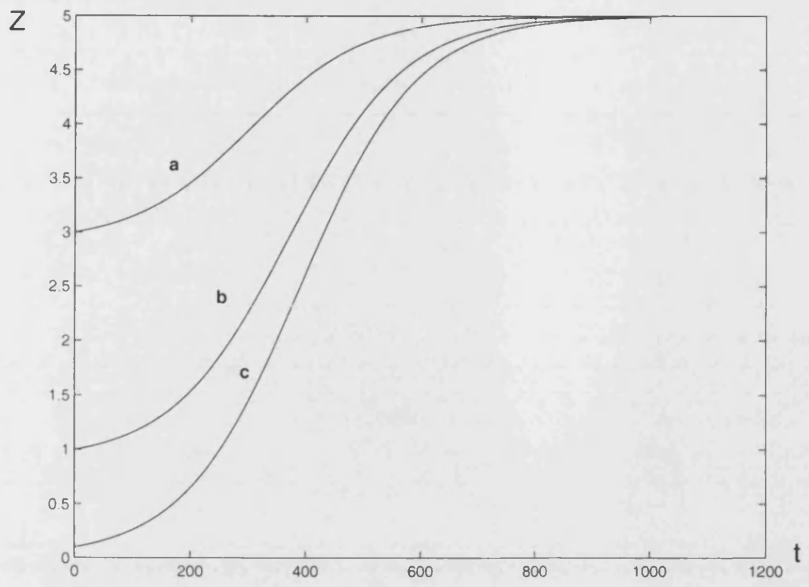



Figure 5.7: Growth functions with different start values, which still have slow initial growth rates.

5.8 shows the rate of change of the growth rate for the curves in figure 5.7. This growth change rate function will provide the description for the force value which applied on the surface control points and will be presented in the later chapters. This force is directly obtained from the growth function.

## 5.5 The Advantages of the New Growth Function

The essential differences between our growth function and the general growth function [Edels88] are as follows: firstly, our growth function can keep the proper growth feature in any growth cycle and take the growth point from the last cycle as the start growth value for the current cycle. This means that our growth function can have continuous growth and the whole growth curve for all growth

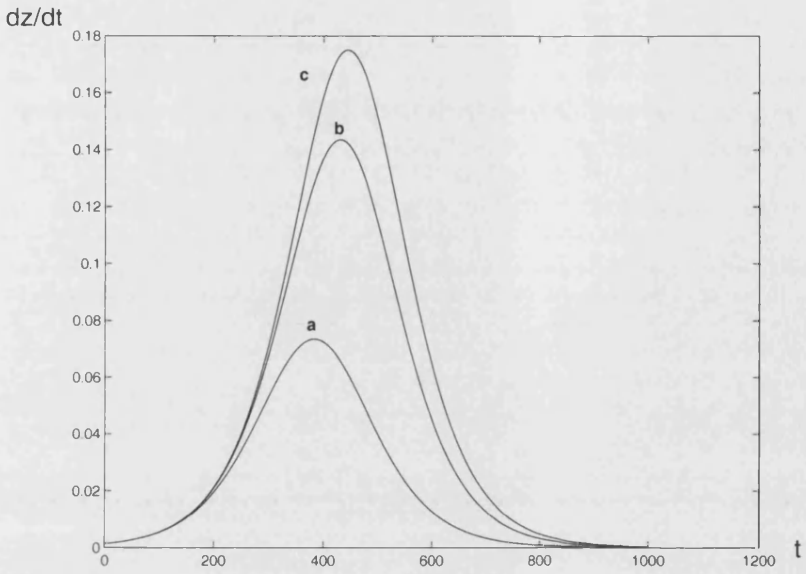



Figure 5.8: Growth change rate functions for figure 5.7

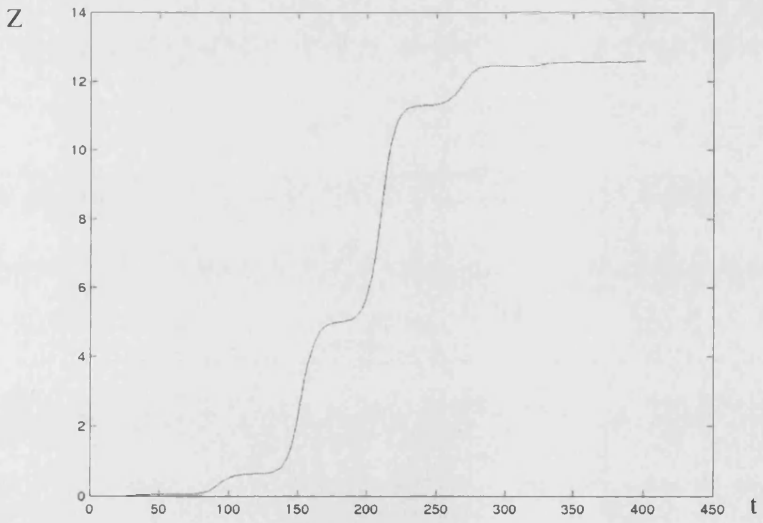



Figure 5.9: Growth curves in different cycles with smooth connection

cycles will still form a general growth function. Secondly, the general growth function has a fixed growth tendency, while the one presented here can have various growth tendencies by using different growth equations. This means that our growth function is changeable. Thirdly, the general growth function has no control of the growth rates during a time interval while ours can show and change the values of all the growth rates.

The advantages of our new growth function are therefore:

- The function can be changed at any moment during the development animation, that is, we can change the equation at any time. This means that the growth rate changing tendency and growth acceleration for the flower petal can be different for the same petal in one growth process. It provides the possibility of having unusual petals by breaking the growth tendency.

- The growth change rate is under control with the appropriate curve. This means that the shapes for the flower petal surface are fully controlled by this function. We can generate different petal shapes by setting appropriate growth rates with this function.

- Although the time interval is fixed for its recursive process, the specific time interval could be set with different equations. This means that we can control the adjustments to the growth rate by controlling the time interval. The specific growth process can be recalled by retrieving the time interval when the desired shape is not achieved.

## 5.6 Conclusions

In this chapter we have introduced a new growth theory for a petal surface model, which can simulate the petal surface's natural development. We also developed and explained our growth function for perturbations in growth.

Our growth function allows continuous growth from the last growth cycle. In addition, the presented growth function enables the growth rate to change at any development stage. A flexible growth change rate curve allows variation in the development tendency at a specific time interval, and therefore differs from other animations in which the plant grows according to a single growth function throughout its growth phase.

We believe our proposed growth function will prove useful for flower development modelling. We now start to consider how to apply the growth function to the surface model. We have decided to apply it to the control points, which will be presented in detail in the next chapter.

# Chapter 6

# Mathematical Framework Model

## 6.1 Introduction

Since petals or leaves change in shape and form during their life-cycle, it is important to construct a suitable dynamic surface model to represent this life-cycle. We have discussed the growth function which can be applied to the flower petal surface in the last chapter. Here we investigate how to construct a dynamic surface model which uses the growth function to represent the natural growth, and is suitable for the petal surface represented by bicubic patch.

In recent years, various dynamic surface models have been developed for different applications and simulations. Tu [Tu94] demonstrated a framework for behavioural animation featuring an artificial fish model with some astonishing behaviours. The mechanics of the mass-spring model from the presented framework can be useful in our system. However, the behavioural modelling for fish is different to our requirements, as it focuses on the fish surface deformation while we have to consider the petal surface growth. Terzopoulos [Terzo87, Terzo88] also presented physically-based models for elastically and inelastically deformable ob-

jects. The models are fundamentally dynamic and realistic animation is created by numerically solving their underlying differential equations. The techniques of handling forces, constraints and obstacles can be useful for our system. Again, due to the different nature of a flower petal surface, different techniques and equations will be developed for our surface model.

Another popular dynamic surface model is the classic mesh method. This has been widely used to model complex smooth surfaces, such as those encountered in cloth [Provo95] and human character animation. However, these methods suffer from at least two difficulties. First, the computation is expensive when the number of mesh nodes is very large. Second, it is difficult to maintain smoothness, especially when mesh nodes need to be repositioned as the model builder requires.

Bicubic patches have the potential to overcome both of these problems: they have limited numbers of control points, and smoothness of the model is automatically guaranteed, even as the model is animated. However, as we stated in chapter two, the bicubic patch is not the mainstream representation used by modellers, due to the mathematical formalisms associated with it. Thus, in order to make a bicubic patch surface a good dynamic surface model, we must find a suitable mathematical framework to describe it.

Most traditional methods for computer graphics modelling are kinematic, which means the shapes are compositions of geometrically or algebraically defined primitives. Kinematic models are passive because they do not interact with each other or with external forces. The models are either stationary or are subjected to motion according to prescribed trajectories. Expertise is required to create natural and pleasing dynamics with passive models. As an alternative, we can use active models, such as a physically-based model. Active models are based on principles of mathematical physics, they react to applied forces, to constraints

or to obstacles as one would expect real, physical objects to react.

This chapter describes a physically-based model for animating flower petal growth that is derived from elastically deformable models, but additionally is improved to take into account the non-elastic properties of the petal surface. In this method the petal is first approximated by a deformable network of masses and springs, the movement of which evolves using the numerical integration of the fundamental law of dynamics and growth theory.

We also introduce constraints on the deformation rates of the springs and the growth rates in order to show the force effect, and we take these constraints into account using a low-cost method inspired by the classical dynamic inverse procedures.

We combine the bicubic patches and the mass-spring model with an appropriate and efficient interface to physical simulation for animation.

The use of the mass-spring and force models poses new challenges throughout the production process, from modelling to animation. However, this model frees the designer from worrying about which control points need to move to maintain a smooth, growing surface. With the introduction of a growth function and dynamic forces, the resulting surface has the appropriate motion and physical properties of a real petal. In the following sections, we present the theoretical background of the proposed physically-based model and discuss the techniques from modelling to animation in detail.

## 6.2 Mass-Spring Model

In this section we introduce the background to the model which supports much of the work in this thesis. We also present the motivation and background for

applying the described model in this research.

We must first ask why the mass-spring model is appropriate for modelling a flower petal surface? According to the biological growth principle used in most growth theories [Edels88], the growth status is defined from the dry weight of the plant. So the mass of the plant plays an important part in the growth function. Here, we apply this concept to the flower petal surface, and distribute the whole petal mass between each of the control points evenly.

We begin by defining the model theory. Our model is a bicubic patch, which can be considered as a mesh of four by four virtual masses, each mass being linked to its neighbours by massless springs of natural length not equal to zero. In a regular framework, as figure 6.1, the linkage between neighbours is achieved in three different ways:

- springs linking masses $[i, j]$ and $[i+1, j]$, and masses $[i, j]$ and $[i, j+1]$, will be referred to as "structural springs";
- springs linking masses $[i, j]$ and $[i+1, j+1]$, and masses $[i+1, j]$ and $[i, j+1]$, will be referred to as "shear springs";
- springs linking masses $[i, j]$ and $[i+2, j]$, and masses $[i, j]$ and $[i, j+2]$, will be referred to as "flexion springs".

Under pure shear stresses, only the shear springs are constrained; under pure flexion stresses (i.e. bending), only the flexion springs are constrained; whereas under pure compression or traction stresses (i.e. stretching), only the structural springs are constrained. We only apply structural springs in our model because our petal surfaces are growth surface and it is unlikely that they are under pure shear stresses or flexion stresses.

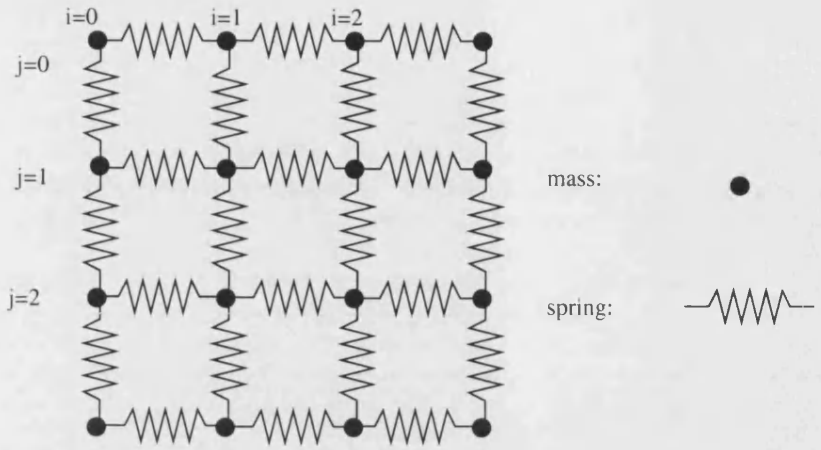



Figure 6.1: Regular framework of masses and springs used for our model

The mass-spring model is a simple technique for representing flexible objects such as cloth, which represents an object with mass-points and springs. The forces caused by the springs make the mass-points move. The springs can be constructed according to different circumstances. This mass-spring model is a very intuitive approach to the representation of surface deformation. A spring responds to a force by deforming elastically in an amount proportional to the force; when the force goes away, so does the deformation [Terzo88]. A grid of such a model deforms and stretches, subject to laws of physics (modelled globally by rigid-body dynamics, but locally by stress and strain rules related to the structure of the units, such as mass-points and springs, in the material in an internal coordinate system).

### 6.2.1 Model structure

Though our application is concerned with petal surface growth instead of surface deformation, we find the mass-spring model structure is very suitable for applying the growth function to our bicubic patch surface.

1. We need to control the control points to enable them to move with the related growth function. The mass-spring model has similar control for its mass-points. So we can easily use control points as mass-points.

2. The most important feature of a plant organ is that it grows, but we must note that this is not equivalent to stretching. It includes size change, shape change and mass change, and maybe even distortions if there is external intervention or any obstruction. With all the links provided by springs, we can reposition the control points on the surface to enable such growth to be animated.

3. The petal surface has mass itself, therefore gravity is one of the forces helping it to open. Gravity is also related to the shape change on the petal surface because it has a direct force effect on the control points. From the theory of growth function, we can take the natural growth as a kind of force to push all the control points apart (such as along the length growth direction or along the width growth direction to have a longer and wider petal surface). Of course, if there is an external intervention, another external force is applied. So all the growth function can be converted to forces applied to the surface. These forces control will match the mass-spring model's principle. Of course, the relevant forces equation will vary with each application.

4. Even though the forces may only apply to one control point of the surface, the growth forces will affect all the linked neighbour points. The plant organ is stiffer than cloth, but it certainly has constraints between all the control points. This structure is similar to a mass-spring model with much stiffer springs.

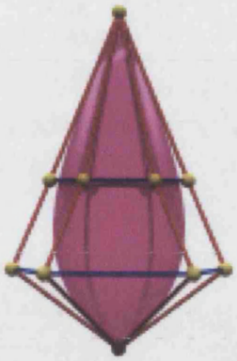


Figure 6.2: Three-dimensional petal surface with mass control points and spring links (Note, apart from the discontinuity points, all control points are above the petal surface.)

From the above discussion, we have sufficient justification to introduce a new method which combines the bicubic patch with the mass-spring model. The specificity of this approach is that the model is not considered as a continuous surface, but instead will have to be discretised as a discrete structure of elements where each mass-point and each spring can be handled individually. In our approach, we start with a model composed of masses and springs, showing how to apply it to the petal surface.

### 6.2.2 Petal surface model description

The bicubic patch is applied here as the surface representation for our flower petal surface but also combined with a mass-spring model. As shown in figure 6.2, the bicubic patch control points are replaced by mass-points, which evenly represent the whole structure of a petal surface. We also have structural springs linking mass-points $[i, j]$ and $[i + 1, j]$, and mass-points $[i, j]$ and $[i, j + 1]$. Any

force applied on a mass-point will have effects on all the other mass-points linked with that point.

At this point, we again emphasise that the springs connect the control mass-points of the bicubic patch. We also emphasise that apart from the corner discontinuity points, all control points are above the petal surface.

## 6.3 Dynamics and Forces

In this section, we investigate how the forces apply at mass-points. The use of physical modelling to animate clothing has been widely discussed in the past literature [Provo95]. For physical modelling, the basic properties of a material are generally specified by defining an energy function to represent the attraction or resistance of the material to a possible variety of deformations. Typically, the energy can be specified as a discrete sum related to forces which are functions of the positions of surface control vertices. Compared to artificially moving the control points to meet the growth requirements, this force controlling model can apply the growth functions, which were converted from the related growth forces, to the control mass-points. This model also considers all kinds of internal and external forces and the constraints of neighbours on the control points. We will therefore modify the traditional equations to deal with our flower growth animation.

The system under study is the bicubic patch of $4X4$ masses, each mass being positioned at time $t$ at the point $P_{i,j}(t)$, where $i = 0, 1, 2, 3$ and $j = 0, 1, 2, 3$. The evolution of the system is governed by the fundamental Newton's law of dynamics:

$$F_{i,j} = u * a_{i,j} \tag{6.1}$$

where $u$ is the mass of each point $P_{i,j}$ and $a_{i,j}$ is its acceleration caused by the force $F_{i,j}$. $F_{i,j}$ can be divided between the internal and external forces. The internal force is the resultant of the tensions of the springs linking $P_{i,j}$ to its neighbours, which are called contact and constraint forces:

$$F_{int-con}(P_{i,j}) = \sum_{(k,l)\in R} K_{i,j,k,l} * L_{i,j,k,l} \tag{6.2}$$

where $R$ is the set regrouping all couples $(k,l)$ such that $P_{k,l}$ is linked by a spring to $P_{i,j}$. $L_{i,j,k,l}$ is the vector from $P_{i,j}$ to $P_{k,l}$, and $K_{i,j,k,l}$ is the stiffness of the spring linking $P_{i,j}$ and $P_{k,l}$.

However, there is another important internal force in our model for the growing flower petal surface, that is, for the growth function of the petal surface. From the growth function, we have the growth rate function which, typically, is slow initially, accelerating to reach a maximum, then slowing again and eventually ceasing. We take this growth into account via the internal force which push the control points away from each other in the growth direction. According to the current growth function, the forces will act in three growth directions, which are related to length, width and curvature growth. This internal force is given by:

$$F_{int-grow}(P_{i,j}) = u * a_{grow} \tag{6.3}$$

where $a_{grow}$ is the acceleration of the growth rate in the time interval $T$. The important feature for these forces is that they are integrated with the growth function. We reiterate that they are small initially, accelerating to reach a maximum, then decreasing and eventually ceasing. At this final stage, there is no growth force and the petal will only deform within the constraints applied by

external forces.

A plant can be subjected to external forces in various ways, such as simple deflection, obstructing forces, periodic forces, etc. One obvious force will be gravity, let $g$ be the acceleration of gravity, the weight of $P_{i,j}$ is given by:

$$F_{gr}(P_{i,j}) = u * g \tag{6.4}$$

Other external forces may come from natural perturbations, such as collisions with other flower petals, and artificial perturbations, such as human intervention. These forces will apply to specific control points which will change the positions of some control points to generate the new growing surface.

In addition, we need to take into account the mass of each control point. In our model, these masses are not constant, but vary with the growth function. This is different from other mass-spring models. Most of mass-spring models are for flexible objects, such as cloth, where the mass normally remains constant during deformation. However, our petal surface is a growth surface, therefore it is obvious that its mass will increase according to the applied growth function until it reaches a maximum value. In the final mature period, the petal surface with maximum mass will be pulled down by gravity and will eventually fall to the ground.

### 6.3.1 Force model implementation

By using the implicit Euler method, the position of each mass point can be updated as:

$$X_i^{t+h} = X_i^t + V_i^{t+h} * h \tag{6.5}$$

$$V_i^{t+h} = v_i^t + F_i^{t+h} * T/m_i \tag{6.6}$$

where $v_i^t$ is the velocity of the $i$-th mass point at time $t$, and $F_i^t$ is the force on the mass point at time $t$. Similarly, $X_i^t$ is the location of the $i$-th mass point at time $t$, $m_i$ is the mass and $T$ is the time interval between each animation step.

And $F_i^{t+h}$ can be approximated with a first order derivative as:

$$F^{t+h} = F^t + \frac{\partial F}{\partial X}\Delta X^{t+h} \tag{6.7}$$

where $F^t$ includes all the internal and external forces described above on the $i$-th mass point, $F^t = [F_1^t, F_2^t, ....., F_n^t]^T$.

From these equations, we can calculate the force value on a mass-point after a specific time interval, the velocity of a mass point at any time and the moving distance or new position following the force applied at the mass point.

## 6.4 Analysis and Performance

In this section, we show how the forces affect the growth of the petal surface.

### 6.4.1 Influence on growth

It is difficult to model biological growth realistically. In our model, realistic growth has been placed in a mathematical framework which models both internal and external influences. The external influences will depend on the direction of external force, which is not easily predicted. However, the main external force is gravity, which also plays an important role in helping the petal open up throughout the whole growth process. The internal forces are integrated with the growth function and applied to three of the main features of the petal, that is,

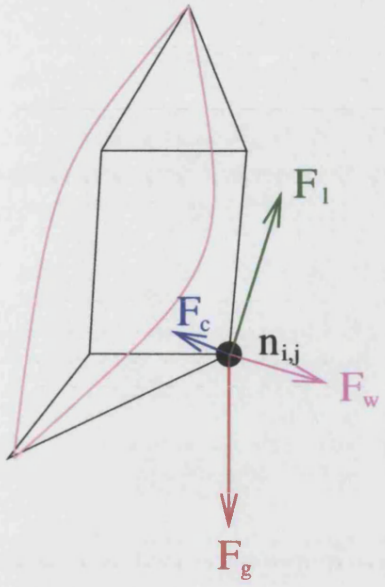



Figure 6.3: Petal forces directions at a mass point

its length, width and curvature. Of course, other internal forces are caused by the constraint forces from the mass-point's linking springs to the neighbouring points.

Figure 6.3 gives an example of one mass-point with all related forces in relevant directions (except the constraint forces which are along the linked springs). $F_g$ represents the gravity, $F_l$ represents the length growth force, $F_w$ represents the width growth force and $F_c$ represents the force from the curvature change.

### 6.4.2 Force effect

Figure 6.4 shows one petal with only the length growth force being applied. In this petal, the length growth causes a force along the length growing direction. As it does not have any other force, we can see the effect from the force will match the growth function. If we draw a curve on all the top points of the petals, it matches the general growth function curve.

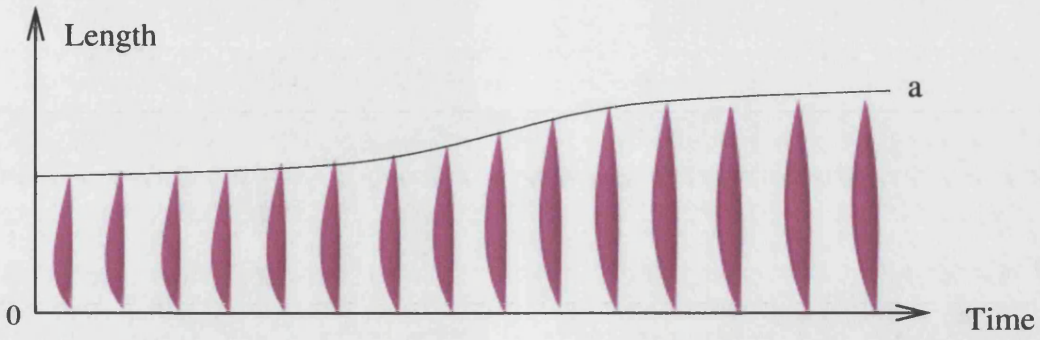



Figure 6.4: Growing petal with only the length growth force applied

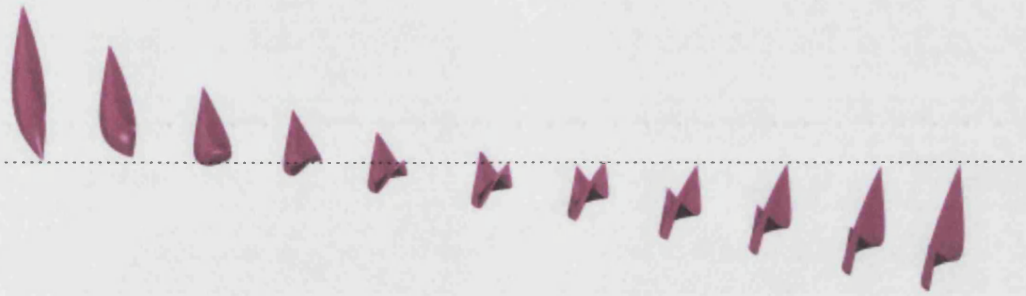


Figure 6.5: Falling petal with only gravity

The figure 6.5 shows the effect from an external gravity force. When there is no other growth force, the mass on the petal surface which is distributed evenly across the control points will cause the petal to fall towards the earth. The mass of the petal is biased towards the centre of the petal. As a result, the downward force is also in this position. The effort on the petal is to distort its shape as shown in figure 6.5. Here, we see that the centre of the petal is pulled down, but the petal resists this motion. therefore, the edges of the petal lag the central motion. As the mass points are constrained to maintain their relative positions, the petal does not stretch. If we do not have this constraint then the petal would artificially stretch under the gravitational force.

## 6.5 Advantages and Comparisons

Though we have discussed the reasons for choosing bicubic patch to represent our petal surface in the previous chapters, the question for surface representation might arise again when we show the combination of our surface representation with the mass-spring model. This is because the mass-spring model always comes with a mesh surface. We investigate this question from comparisons with the cloth model and mesh nodes surface, and show some examples from our surface model.

### 6.5.1 Comparison with the cloth model

It might appear that work done on modelling cloth in computer graphics may be directly applicable to modelling plant components such as petals. However, the models required for cloth do not display the necessary attributes.

The mass-spring model with a mesh has been widely used for cloth model [Provo95]. There are some features for cloth widely used in animation systems. Firstly, the cloth is homogeneous, isotropic and linearly elastic in its initial shape. Secondly, the cloth is in equilibrium at any time under a given applied force. Finally, the cloth is a perfectly thin surface and never expanded or contracted along its surface normal. However, in our petal surface model, the length, width and curvature changes will expand the surface in all growing directions under the constraints.

Thus, despite the fact that the cloth mesh model may appear to have attractive results in animation and simulation systems, we must choose an alternative appropriate surface to represent petals and leaves.

### 6.5.2 Comparison to a mesh surface

Our model combines a bicubic patch surface with a mass-spring model. We must notice that the springs connect the control mass nodes of the bicubic patch and do not connect mesh nodes on the petal surface. This difference comes from the advantages of our bicubic patch method applied to the mass-spring model.

Compared with the mesh surface model, our bicubic patch method has the following advantages:

- The control points can be used to generate a growth surface. With the mass-spring and force model, the sixteen control points on the bicubic patch have all the growth information and can generate a smooth surface. While a simple mesh with few points may be sufficient for a small initial surface, more points must be added to that mesh when the surface grows larger. This means that a dynamic mesh is necessary for this kind of growth surface to remain smooth. As shown in figure 6.6, the initial mesh is enough to represent the small petal shown on the left, but it is not enough to describe the surface details when it grows to the larger surface shown on the right. The initial mesh must be subdivided as shown in the left corner of the larger petal. This dynamic mesh will also have to pass all the growing and control details to the sub-meshes. As the petal grows, more mesh nodes are needed for the surface to remain smooth. An example of this can be seen in figure 6.7. If each mesh node has all the related force and growth information and some points need to be adjusted according to external forces which will involve more points being repositioned, the computation cost will make real-time animation difficult.

- The cost of computation is reduced dramatically. As a real-time simulation,

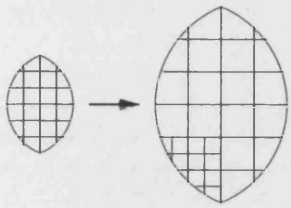


Figure 6.6: Mesh surface for growing petal

Figure 6.7: As the plant grows, more mesh nodes are needed for the surface

Figure 6.8: Petal surface with mass control points

it is very important to have all the information calculated for the model with a minimum computation cost. Even though the bicubic patch contains much growth information, the limited control points make it possible to cut the computation cost. However, with a mesh surface for the growing petal model, the extra growing force (including the internal forces from its neighbour points and the external forces) must be considered for each mesh point. All force vectors will add up to very costly algorithm even before we take account of the fact that the new dynamic mesh points might need more information.

### 6.5.3 Results from our model

As we can see from the figures 6.8, 6.9 and 6.10 (petal surface with mass control points), our method can generate a smoothly curved surface with only a few forces applied to the control points. However, if those control points are nodes on a traditional mesh, they cannot form a smooth surface without the key curvature points. Many extra mesh nodes must be added to the current mesh, each containing details of forces and growth information, which is difficult for real-time

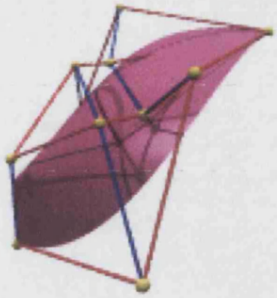


Figure 6.9: Bending petal surface with mass control points

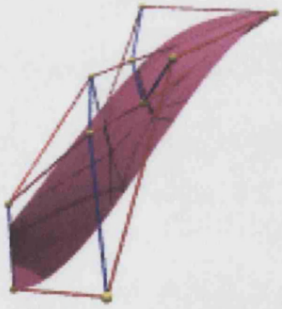


Figure 6.10: Growing petal surface with mass control points

animation. Because of these difficulties we reject the traditional mesh method as inappropriate for animating petals.

However, the combination of bicubic patches and the mass-spring model delivers a very efficient method to represent the growth of a petal surface. In summary, our model provides a direct method of surface control, which is more flexible and creative,while at the same time requiring considerably less computation than a traditional mesh.

## 6.6 Collisions

Collision and deformation for free form surface modelling remains a challenging area because of the flexibility on the surface. Collision handling is an important task for any growing surface model, and must include both collision detection and collision response.

Simulation of collisions between mature organs is an important problem in the visualisation of structures with densely packed organs such as flowers. In nature, individual flowers touch each other, which modifies their positions and shapes. Consequently, the mature organs must be carefully modelled and sized to avoid intersections. This is feasible while modelling static structures, but proper simulation of collisions would become crucial in the realistic animation of plant development. This technique will be discussed in this section.

### 6.6.1 Collision detection

The collision detection of free-form surfaces and three-dimensional objects is a very difficult problem in geometric computation. The calculation of collisions, however, can be extremely time-consuming for a computer system, and our sys-

tem therefore allows the user to select whether or not to calculate collisions for a given object.

There are different ways to carry out the collision detection. The simplest approach to detecting collision in a physical simulation is to test each geometric element, such as point, edge, face, against each other geometric element for a possible collision. To achieve practical running times for large simulations, the number of possible collisions must be culled as rapidly as possible using some type of spatial data structure. If the surface connectivity does not change, a two-dimensional surface-based data structure can be used for the elements. Another method is to distribute the elements into a three-dimensional volume-based data structure. However, it is too expensive in the computation and subdivision of the bounding boxes of all the elements, especially when all the elements are surfaces. This is because surface intersection has a very high computation cost and it is not easy to balance the three conflicting goals of accuracy, robustness, and efficiency.

A simple real-time collision detection method is described by DeRose [DeRos98]. This method tests every geometric element against every other geometric element for a possible collision using some type of spatial data structure. It is also a simple matter for us to construct a suitable surface-based data structure for the petal surface. A hierarchy construction will be developed with the original convex hull property method [Lu01].

So our method will focus on how to construct a suitable surface-based data structure for the petal surface. As each surface patch has sixteen control points, we can use each group of four control points as a data structure. An efficient way for constructing the hierarchy of boxes is to compute the bounding surfaces using the convex hull property.

If there is no intersection between the two horizontal convex polyhedron (from

the two middle sets of control points) and the vertical convex polyhedron, there is no collision between the two adjacent surfaces. If there is an intersection, it does not imply that a collision has occurred, the surfaces maybe has collision or maybe not. We have to calculate the exact surface points to find out the result (we could use forward differencing techniques to calculate the points).

### 6.6.2 Collision response

After the collision is detected, we must also consider how to deform and arrange the surface to avoid further intersections.

An underlying model [MacCr96] can be deformed by establishing positions of the points of the model within the converging sequence of lattices and then tracking the new positions of these points within the deformed sequence of lattices.

An efficient deformation process must be developed for the free-form surface. Firstly, the control points moving area is established from the above analysis result. That means we have a specific space for the control points, which makes it possible to search the optimisation position for the control points. Secondly, the optimal method should be applied for efficient deformation for the whole group of control points. Thus we need to develop a technique or function for all the control points. Thirdly, surface smoothness feature and growth principle should be met, which are also main concerns of the considerable elements from the collision analysis.

From the collision analysis and growth direction of the control points, we are going to suggest a more efficient deformation method to respond the collision. In that way, we could predict that the control points will form a collided surface and advise to avoid the collision, which makes it possible to have a natural growth

surface. We will discuss how to implement this in the next chapter.

### 6.6.3 Summary

The improvements shown in this section are: The collision detection algorithm is more comprehensive and can be applied to Bézier surfaces for other applications, when surface intersection points are not required, in general. The algorithm is designed to be more efficient for interactive free-form modelling. In addition, the completed deformation process takes account of natural perturbation and surface feature.

## 6.7 Conclusions

In this chapter, we reviewed the concept of the mass-spring model and presented a new combined method for better surface control. We also investigated the integration of growth theory, the control mass-points, and forces control with the mass-spring framework, to enable the surface to grow according to biological growth theory.

Compared to conventional Bézier or B-spline patches, our model has more direct and visible control for the surface; compared to a dynamic mesh model, our model needs much fewer control vertices to obtain a desired shape. These features are quite attractive, especially when this model is adopted for animation work, during which vertices are moved with time and yet smoothness of surfaces for each animation frame must be maintained. Since the surface is represented by few control points, local control can be easily obtained by repositioning the control vertices and the generation of the surface is simpler and faster than other techniques such as implicit surfaces. When the surface responds to growth, only a

few related control points need recalculation. Consequently, a substantial reduction in computation time is achieved. This feature is likely to make it a promising model for real-time natural animation.

We concluded this chapter with the presentation of collision detection and collision response. In the next chapter, we continue the important discussions of surface collisions and the related techniques for collision avoidance. We will introduce the optimisation techniques genetic algorithms for our surface control.

# Chapter 7

# Genetic Algorithms for Surface Control

## 7.1 Introduction

It is often difficult to build interesting or realistic virtual entities and still maintain control over them. Sometimes it is difficult to build a complex world at all, if it is necessary to conceive, design, and assemble each component. An example of this trade-off is that of kinematic control versus dynamic simulation. If we directly provide the positions and angles of moving objects, we can control each detail of their behaviour, however, it might be difficult to achieve physically plausible motions. If, instead, we provide forces and torques that simulate the resulting dynamics, the result will probably look correct, but then it can be very difficult to achieve the desired behaviour, especially as the objects we wish to control become more complex. Methods have been developed for dynamically controlling specific objects for movement in character modelling, but a new control algorithm must be carefully designed each time a new behaviour or morphology is desired.

As we described in the last chapter, the surface shape is converted into a mathematical framework with the controls from internal and external forces. However, these forces applied on the surface may not generate a natural surface, particularly if the surface is in collision. It is not very realistic to require the user to adjust the growth force to avoid all collisions, which means that the control points of the model must be placed and calculated very carefully. Fortunately, as an alternative, optimisation techniques offer possibilities for the automatic generation of complex systems. The genetic algorithm is a form of artificial evolution, and is a commonly used method for optimising complex systems. A Darwinian "survival of the fittest" approach is employed to search for optimal evolution in large multidimensional spaces. The use of genetic algorithms can permit virtual entities to be created by the user without requiring an understanding of the procedures or parameters used to generate them. These procedures and parameters, implemented by the system creator, remain hidden from the user. The measure of success, or fitness, of each individual entity can be calculated automatically, or it can instead be provided interactively by a user.

Genetic algorithms are search algorithms based on the mechanics of natural selection and natural genetics [Goldb89]. They combine survival of the fittest among string structures with a structured yet randomised information exchange, to form a search algorithm with some of the innovative flair of human search. It means that the same algorithm may have various results according to different search rules and human stylishness.

Genetic algorithms can be used to explore the space of possible animations of a character model. The objective of a character animator in these methods is the specification of a controller for a character. They have proven useful for searching large spaces using simulated systems of variation and selection [Goldb89]. In

Sims's work [Sims94], genetic algorithms have been used for the generation of controllers in physically modelled articulated figures.

The model we described in the previous chapter is much more flexible and controllable than most models reported in the literature. We control all growth by applying the appropriate forces. This is different from most previous work where the dynamic model of the simulation is predetermined by having the same procedural method controlling the forces. The obvious disadvantage of this latter approach is the loss of flexibility in the animation. Conversely the methods described here based on genetic algorithms are highly flexible.

Obvious, the user will sacrifice some control because the fitness function is procedurally defined. However, the potential gain in automating the creation of complexity can often compensate for this loss of control, and a higher level of user influence is still maintained by the specification of the fitness criteria. In addition, the involvement of the genetic algorithm is evolutionary and optimises the growth mode. In this chapter, we start with related work and then discuss how to combine genetic algorithms with our previous model, to avoid petal surface collisions.

## 7.2 Related Work

Genetic algorithms have been used recently in some computer graphics areas. For example, they have been used in free-form surface matching [Brunn97] and character animations [Zakar01, Rana95]. The applications using genetic algorithms are always associated with the real world and dynamic environment modelling. In this section, we review some other methods used in artificial life animation and present the related usage of genetic algorithms in the similar area of real

dynamic environment animation. The comparison will show the advantages to be gained from using genetic algorithms and explain why we have chosen to use them in the work presented here.

### 7.2.1 Animated artificial life

Computer simulations occurring in virtual spaces of computer memory enable great artistic flexibility and supply a context for basic research in adaptive behaviour. A number of physically based locomotion systems have been designed for animation to achieve realistic animal motion using forward dynamics, rather than kinematic methods. Early examples of using artificial life principles in computer animation include Reynold's distributed behavioural model [Reyno87], in which collective behaviours (flocking, herding, etc.) emerge from many interacting agents. Other search techniques for physically based modelling include space-time constraints [Liu94], which automatically generate the motion over the whole animation sequence at once. This method is optimised in both space and time, but is computationally intensive and complex. In addition, the motion is entirely specified. For example the end target position cannot be modified without rerunning the entire simulation. Simulated annealing is also used to search for a globally optimal solution. It uses a temperature parameter to determine the chance that a change in a parameter will be allowed to worsen the optimisation. However, its drawback is that it depends on a control system model with a fixed set of parameters to optimise [Ngo93]. So the topology of the control network is fixed and only its parameters change. The choice of these parameters is critical, and will decide whether the problem is soluble at all.

### 7.2.2 Genetic algorithms in animation

Genetic algorithms have been used for evolving goal-directed motion in physically-based animated figures, including a technique for evolving stimulus response mechanisms for locomotion [Ngo93]. The stimulus is a function of sensors in the character, and the response is a list of desired angles for all the joints in the creature. A complete model of fish locomotion with perception, learning and group behaviours, which generates very realistic animations, was developed by Tu and Terzopoulos [Tu94]. The virtual creatures of Sims [Sims94] are well-known examples of artificial life entities. Sims has developed a comprehensive physical and 3D shaded model for defining a creature genetically. Creatures with controllers evolved by genetic algorithms should allow more complex and realistic models from biology than have been previously explored. Genetic algorithms have also proved particularly useful in our flower growth animation.

## 7.3 Model Description

### 7.3.1 Basic model

A genetic algorithm used to solve an evolutionary problem must have five components:

1. a string representation of solutions to the problem (called a genome);
2. a way to create an initial population of solutions;
3. an evaluation function that plays the role of the environment, rating solutions in term of their "fitness"; this is usually called the fitness function;

4. genetic operators that alter the composition of control factors during reproduction;

5. values for the parameters that the genetic algorithm uses (population size, probabilities of applying genetic operators, etc.).

### 7.3.2 Fitness function

A genetic algorithm needs a cost function or fitness function to be defined for a given problem. It is an evaluation function to decide if the search result is fit for the application. In this section we develop fitness function criteria appropriate for our surface controlling problem.

In genetic algorithms the choice of a fitness function needs careful consideration. The convergence of the algorithm depends crucially on the relative search size of this whole solution space. A choice we make is to build up a cost function that has some meaning, in particular we aim to find an intermediate qualitative measure which counts the number of good controls. This will then be transformed into a function which will be the fitness passed to the genetic algorithm.

To build a simulated growth, we start with a model that supports geometric structure. From the analysis of the geometric structure, we can define the criteria for the fitness function of the petal surface model. We add behaviours that correspond to the opposing forces of collision avoidance and the urge to grow. The behaviours that lead to simulated growth for all the petals are:

1. Collision avoidance: avoid collisions with nearby petals as the petals grow and change shape;

2. Velocity matching: attempt to match velocity with nearby petals when they are open together;

3. Flower centring: the bottom part of the petals attempt to stay close to nearby petals and the top part of the petals attempt to open along the line to the flower centre.

Velocity is a vector quantity, referring to the combination of direction and speed. Static collision avoidance and dynamic velocity matching are complementary. Together they ensure that the petals of one flower are free to open within the constraints and growth functions without colliding with one another. Collision avoidance is the urge to steer away from an imminent impact. Static collision avoidance is based on the relative position of the petals and ignores their velocity. Conversely, velocity matching is based only on velocity and ignores position. It is a predictive version of collision avoidance: if the applied force does a good job of matching velocity with its neighbours, it is unlikely that the petal will collide with any of them at any time in the future. With velocity matching, separations between petal control points remains approximately invariant with respect to evolving geometric structure. That means the separation distance between petals remains in the geometric structure for the next time step frame. In other words, static collision avoidance serves to establish the minimum required separation distance between petals, whereas velocity matching tends to maintain it. Flower centring is another important rule to keep all the petals open within the constraints of the flower boundary and also provides the opening directions for the flower petals.

Genetic algorithms are general optimisation algorithms with a few components and do not specify an approach for any application and implementation. We therefore have to define the search rules and fitness functions according to our application and problems. All the behaviours, rules and criteria presented above

lead to the discussion of how to implement the genetic algorithms for our petal surface model.

## 7.4 Model Implementation with Genetic Algorithms

### 7.4.1 Parameters to represent the problem

In this work we have chosen to use genetic algorithms to solve the problem of collisions avoidance between flower petals. The problem is if we allow the flower petals to grow with control parameters designed by the user, the petals are likely to collide with nearby petals. It is obvious that the solution is to adjust the parameters in the growth function while keeping the petals growing. As described in the last chapter, we convert the growth function to forces controlling a mass-spring model, thus we need to adjust the forces for the petal growth and collision avoidance.

As a result, the parameters required to represent the collision problem are the same as those controlling the surface change, which related to the changing forces applied to the surface control points. In our implementation, these parameters are length, width, surface curvature and flower opening speed.

### 7.4.2 Growth values for the parameters

We know that genetic algorithms work with a coding of the parameter set, not the parameters themselves. Thus we have to investigate how to encode the parameters. In order to find out the probabilities of applying genetic operators to the parameters, we need to provide the values for the parameters that genetic al-

gorithms use. In this way we will be able to create the possible growth directions for all the control points in our petal surface model.

Growth simulation in Dobashi's work [Dobas00] creates realistic cloud motion with only a relatively small amount of computation. Dobashi uses cellular automata that can simulate the motion just by simple Boolean operations. Three logical variables are assigned to each cell. The state of each variable is either 0 or 1. The value of the Boolean function is calculated by the status of action around the cell. Most of the functions relate to the fact that clouds grow upward and horizontally. Dobashi's work is an efficient simulation method for realistic animation of clouds.

In our work we use a similar principle for encoding the growth parameters. However, it should be recognised that our petal surface growth control is more complex than the cloud growth presented by Dobashi. A Boolean function is insufficient to represent the petal surface growth and respond to collisions interactively. This is the reason we need genetic algorithms for surface growth and collision response. However, the Boolean operations provide a simple encoding method for the surface model. In our system, we set the value of 0 or 1 for the forces parameters, which are in different directions according to the related growth parameters. The value 1 indicates that the growth can continue in its direction and the value 0 means the growth will not happen in that direction. For example, if in a specific time step, a control mass-point has growth in length, width, surface curvature and flower opening direction, the coding value for all the forces parameters applied on that point is 1111. However, if the petal will incur a collision if it continues to grow wider, we might like to have forces parameters code as 1011, that means the width growth will not happen. It is not always clear which factors will eventually cause a collision. Thus, searching for a suitable code

is the task for genetic algorithms with the consideration of fitness function and changing codes is the task for genetic operation. This approach will be discussed in the following sections.

### 7.4.3 Fitness function

The fitness function determines if the forces applied with the code meet our criteria of natural growth without incurring collision. The fitness function varies with different applications and practical problems. In our application, we mainly consider the three requirements for realistic growth described in the last section: collision avoidance, velocity matching and flower centring. We have discussed in the last section that these criteria are sufficient for generating natural flower growth.

The fitness function attempts to place a penalty on the movement of the surface that does not meet the criteria for collision avoidance. Different aspects of the fitness function are described below:

(1) The collision penalty depends on collisions between petals in corresponding points on the petal surface model, thus the equation for collision avoidance factor $S_1$ is given by:

$$S_1 = \sum_{i=1}^{N} C_i \tag{7.1}$$

where $i$ represents all the points on one petal model, from 1 to $N$. $C_i$ represents the penalty on point $i$. It is 1 if there is collision, and 0 if there is not.

(2) The distance moved in a given direction during a unit of time determines the velocity. Thus velocity matching factor $S_2$ is given by:

$$S_2 = \sum_{i,j=1}^{N} |D_i - D_j| \tag{7.2}$$

where $D_i$ is the distance that point $i$ travel during a time interval, and $D_j$ is the distance that point $j$ travel during the same time interval. We must emphasise that point $j$ is not from the same petal as point $i$, instead it is from the adjacent petal.

(3) Flower centring factor $S_3$ is given by:

$$S_3 = \sum_{i=1}^{N} |V_i \bullet V_c| \tag{7.3}$$

where $V_i$ is the vector at point $i$, with the direction from the previous position pointing to the new position after the time interval. $V_c$ is the vector along the horizontal line going through the flower centre (see figure 7.1). It is obvious that $S_3$ will be 0 if these two vectors are perpendicular.

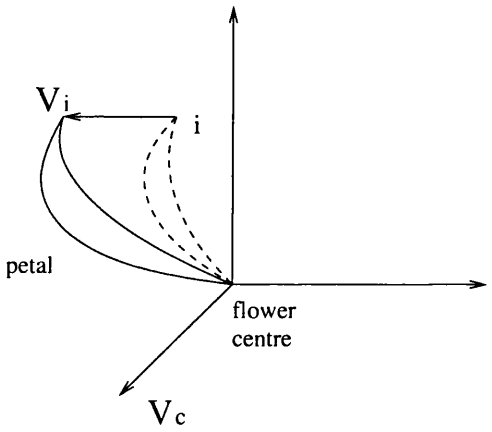



Figure 7.1: Flower centring vectors on growing petal

So the fitness function $F$ is given by:

$$F = k_1 * S_1 + k_2 * S_2 + k_3 * S_3 \tag{7.4}$$

where $k_1$, $k_2$ and $k_3$ are constants. The value of $k_1$ is kept relatively higher, since collisions are to be avoided at all costs. The value of $k_3$ on the other hand is

kept very low, since normally the force already can keep the petal moving along the flower centre direction. It is obvious the algorithm should try to search the solution space for the lowest $F$ value for petal growth.

### 7.4.4 Genetic operation

The basic genetic operations for the genome include reproduction, crossover, and mutation. The reproduction operator determines how the genome is initialised. It is formed by the order of a list of parameters from the forces applied on the specific point on the petal surface model. The mutation operator defines the procedure for mutating each genome. A typical mutator for a binary string genome simply flips the bits in the string (see figure 7.2) with a defined probability. In our approach, we randomly perturb the control points force for opposite growth direction. That is because if there is a collision, an opposite force will be generated in the push the point to opposite direction. The crossover operator defines the procedure for generating a child from two parent genomes. Here, we use a standard single-point crossover to generate offspring (see figure 7.3). Given two individuals (possible solutions to a problem) in a population, the genetic algorithm generates a child solution by randomly taking some of the genes from one individual, and taking the remaining genes from the other.

| | |
|---|---|
| before | 1 0 1 1 0 0 0 1 1 1 1 |
| after | 1 0 1 1 0 1 0 1 1 1 1 |

Figure 7.2: Example of mutation in binary encoding

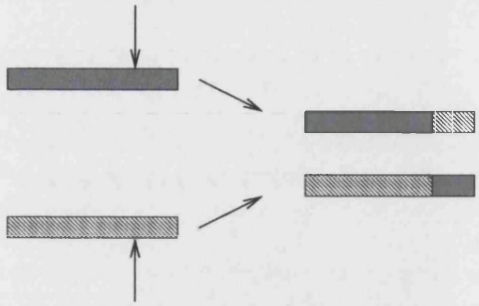


Figure 7.3: Single point crossover

## 7.5 Comparison

In this section, we compare the results from genetic algorithms with those results without the implementation of genetic algorithms. A traditional method without a space search optimisation algorithm will determine forces relative to the growth function chosen by the user. However, this growth force might move the control point too much and thereby generate a surface in collision with the neighbour growing surface. Thus the modification of the force is important for collision avoidance.


Figure 7.4: Flower grows looser with collision

### 7.5.1 Flower growth result

We can see from Figure 7.4 that as the flower grows majority part of the three petals cross each other from status ($a$) to ($b$) if we allow the flower to grow with the parameters as set by the user.

However, with the genetic algorithm, the forces are changed at each time step during the growing period. The changes to the forces in both value and direction. The criteria of changes are determined by the fitness function, and the rules of changes are from the genetic operations presented in the last section. Figure 7.5 shows the effect of using a genetic algorithm. Here at each stage of growth the petals have maintained a tight formation but without overlap of any petals. Without the use of the genetic algorithms this result would have been difficult to derive, even if the user modified the growth parameters at each time step. With the use of the genetic algorithms the forces have been automatically modified at each time step to ensure that overlapping petals cannot occur.


Figure 7.5: Flower grows tighter and avoids collision

## 7.6 Advantages and Comparison

Genetic algorithms (GAs) are different from more normal optimisation and search procedures in four ways:

1. GAs work with a coding of the parameter set, not the parameters themselves;

2. GAs search from a population of points, not a single point;

3. GAs use objective function information, not derivatives or other auxiliary knowledge;

4. GAs use probabilistic transition rules, not deterministic rules.

So they can help us not only optimising the petal growth space, but also generating evolutionary natural petal surface.

The advantages of applying genetic algorithms are:

1. The absolute fitness functions can be defined by the user. This allows automatic selection of fit individuals and allows large populations to evolve without user intervention. Then we might generate different flower patterns.

2. It is particularly useful for simulating biologically realistic creatures in virtual environment, because the fitness is measured relative to each other which shows competition with other creatures and interaction with the environment in natural settings. So it is easily to match the flower growth functions.

3. The search is also guided by human creativity, through the choice of fitness functions to produce more interesting and acceptable results. So in some views, the user still has control over the flower shape.

4. The search space does not have to be limited, genetic algorithms can combine different factors in a control system. The use of a variable fitness function also allows for the design of a robust controller. This means we can add more controlling parameters for encoding the controller to enable a larger search space.

The summation of these advantages has enabled a flower to develop in a realistic manner.

## 7.7 Conclusions

In this chapter, we presented the theoretical background of genetic algorithms, and described a novel system for creating virtual flowers that grow and behave in simulated three-dimensional physical worlds. The modifications of the growth controlling forces on the surface are generated automatically using genetic algorithms. Suitable fitness evaluation functions are developed to guide the simulated evolutions towards specific interventions such as collisions and artificial interruption.

The goals of this system are: (1) to abstract and rigorously explain the adaptive processes of natural growth systems, and (2) to design artificial systems software that retains the important growth features of natural systems. This approach will be of benefit in both natural and artificial systems science. We concluded the chapter with the advantages for adopting the genetic algorithm approach and the improved realistic results from the growth animation system.

# Chapter 8

# Conclusions and Recommendations

## 8.1 Conclusions

Throughout this thesis, there is an underlying assumption that the natural flower growth life-cycle is important for virtual environment animation. In this research, we have concentrated on the quality of evolutionary and realistic results from different aspects, varying from surface representation models to collision avoidance techniques.

We started with the model design requirements, which must be able to simulate a biological life-cycle for the plant. In chapter 2, we analysed and compared the advantages and disadvantages of the representation methods for surfaces with the objective of using these surfaces for modelling flower petals. From the considerations of mathematical, computational and complexity analysis, we chose the bicubic Bézier patch for our perceptual realistic surface model.

Flower growth animation can never be realistic without consideration of the

positions and arrangements of petals and seeds. With the comprehensive reviews of the phyllotaxis and Fibonacci theory for plant organs in chapter 3, we built the spiral phyllotaxis model for the flower head. This gave a biological rule for how the positions are achieved for the seeds on the flower centre and also the positioning of petals. Phyllotaxis methods were then used in all subsequent work involving the positioning of petals.

Petals are the most important components in our flower model. It is crucial to have a method for generating flower growth animation in which the petal surface and shape can be changed simultaneously in real time. In chapter 4, we presented the petal surface model with easy access to individual control points. In addition, we described various kinds of surface growth control, step by step with analysis of the petal shape changes, and illustrated the use and results of some growth factors in some common cases.

We introduced a growth theory for the petal surface model in chapter 5, which can simulate the petal surface's natural development. The advantage of our growth function is that it allows continuous growth from the previous growth cycle. In addition, a flexible growth change rate curve allows variation in the development tendency at a specific time interval, and therefore differs from other animations in which the plant grows according to a single growth function throughout its growth phase.

The result shows the proposed growth function is useful for the flower development modelling. Thus we needed to investigate how to apply the growth function to the surface model. In chapter 6, we introduced a combination method for better surface control. A promising model for real-time natural animation is generated by combining the mass-spring model with the bicubic patch surface. The integration of growth function, the control mass-points, and forces control

with the mass-spring framework enables the surface to grow according to biological growth theory. This is a new method that has the benefit of a physical model, that obeys the laws of dynamics, being in combination with a surface model that has easily controllable shape and growth features. In addition, the physical model gives the user access to all the dynamic parameters involved in motion control. In particular, being able to calculate dynamic forces will give more realism to the animation.

Compared to conventional Bézier or B-spline patches, our model has more direct and visible control of the surface; compared to dynamic mesh models, our model needs far fewer control vertices to obtain a desirable shape. These features are attractive, especially when this model is adopted for animation work, during which vertices are moved with time, and yet smoothness of surfaces for each animation frame must be maintained. Since the surface is represented by few control points, local control can be easily obtained by repositioning the control vertices, and the generation of the surface is simpler and faster than other techniques such as implicit surfaces. When the surface responds to growth, only a few affected control points need recalculation. Consequently, a substantial reduction in computation time is achieved. This feature is likely to make it a realistic model for real-time natural animation.

The combination use of our model allows model builders to arrange control points in a way that is natural to capture geometric and biological features of the model, without concern for maintaining the smoothness and continuity of the surface. This freedom has two principal consequences. First, it dramatically reduces the time needed for controlling the surface. Second, and perhaps more importantly, it allows the initial model to grow in such a way that the evolution of elements, such as petals, can be accurately and realistically represented and

controlled. Thus, by developing the mass-spring bicubic patch for physical-based surfaces, we have removed two important obstacles found in classic mesh models. By introducing the force function connected to the growth function, we have made this surface model the choice for our petal growth simulation.

At the same time, we can not ignore that facts that individual flowers and their petals touch each other in nature. Simulation of collision between mature organs is an important problem in the visualisation system. After the discussions of collision detection and response in chapter 6, we determined to find a solution to avoid the collision beforehand, especially when we have so much control over the petal surface, flower structure and growth function. Thus, we introduced genetic algorithms in chapter 7 to optimise the controlling forces on the petal surface to enable the growing petal to avoid collisions. A suitable fitness function has been developed to guide the simulated evolution towards specific internal and external interventions. The ability to use genetic algorithms was a direct consequence of developing the mass-spring model in chapter 6. Without access to the dynamic forces involved in collision detection and collision avoidance, we would not have been able to build a relevant fitness function. Therefore, there is additional benefit in adopting the combined surface and mass-spring model.

Many previous models of plant growth avoid the modelling of surface detail, often resorting to imposing detail such as leaves and petals from a library of simplistic elements. Here, we have demonstrated that surface detail can be effectively represented by dynamic elements that grow and deform naturally as the plant evolves through its life-cycle.

In summary, we believe that the proposed modelling method and its extensions will prove useful in many applications of surface modelling, from research in plant development and ecology to the surface design of plant organs and in

the production of animated surface models for use in virtual environments.

## 8.2 Contributions of the Thesis

This thesis consists of two parts: one theoretical and one practical. In the first part, we analyse the surface model and the biological growth rules. In the second part, we apply the improved model for the flower growth animation. The following lists summarise our research contributions from these two activities:

1. Expand the spiral phyllotaxis theory from the flower centre to the arrangement of the flower petals, and implement it.
2. Review and modify the growth function for our growth model.
3. Combine the mass-spring model with bicubic patch for the growing surface.
4. Apply the force model on the surface control points with the biological growth function and genetic algorithm fitness functions.
5. Surface collision detection and the following response and collision avoidance using genetic algorithms.
6. The combinations of all the above for our interactive flower growth animation.

## 8.3 Recommendations

There are a number of aspects to the techniques presented in this thesis which could be developed further or be linked to other research areas:

1. In order to allow non-expert user to have direct and intuitive control for the flower petal growth, we only included some important growth factors, such as length, width and curvature change, in the growth function. More factors and their growth directions could be considered and investigated to provide more complex control. A more detailed model could be developed for more growth factors and directions.

2. As we mentioned in chapter 4, our petal surface model only uses one bicubic patch. It is suitable for the majority of leaf shaped petals. However, research into multi-patch surfaces is suggested for more complicated petal shapes, such as non-divided petals and pollinated shape petals (as in many orchids). Combining a multi-patch model with the mass-spring model will introduce some interesting technical problems at the interface of each petal. The solution to this problem will require further research.

3. In our growth model, the user can choose a growth function by setting the growth minimum value, maximum value and growth change rate. This is a rather mathematical description for the growth function curve. If these ideas are linked and applied in biological research, we suggest that the biologist makes this mathematical model more meaningful by connecting all the constants and parameters to biological terms, such as temperature, sunlight index and DNA. It is similar for controlling forces applied on our petal surface model. In the biological area, more internal forces, such as the one between cells, could be added and explained to meet biological needs.

4. As our model is designed for perceptually realistic growth animation, the collision detection started with a simple convex hull test. To improve the accuracy of the solution or quality of the collision detection, a high com-

putation cost surface detection could be used, especially in non real-time animation applications.

5. Genetic algorithms are optimisation search algorithms. They are defined by distinguished components and genetic operations, not by the fitness function. The fitness function we applied is just one solution. A library of fitness functions can be built for different kinds of search solutions or results. The collision avoidance methods developed may have more general applicability, particularly in figure animation or even in robotics.

## 8.4 Final Words

Carrying out this research has led us to pose new problems for which solutions have yet to be found. But in building this animation system we have demonstrated that adaptability can be combined with a mathematically correct physical model. Therefore, we believe that we have shown a path towards building a more comprehensive system in the future.

# Appendix A

# Implementation

## A.1 Implementation Procedure

### A.1.1 Procedure

The main procedures of the animation system are:

1. Choose an initial petal shape for a bud, with the the control points matrix for $x, y, z$ axis individually. The system also has default data.

2. The user inputs in all the growth rate information and requirements for the petals and flower centre, with all the animation and rendering preferences.

3. The system will generate all the frames of the flower shapes according to the user design. The user can connect all the frames to flower growth animation.

### A.1.2 User Interface

The system user interface is shown in figure A.1. The details is as follows

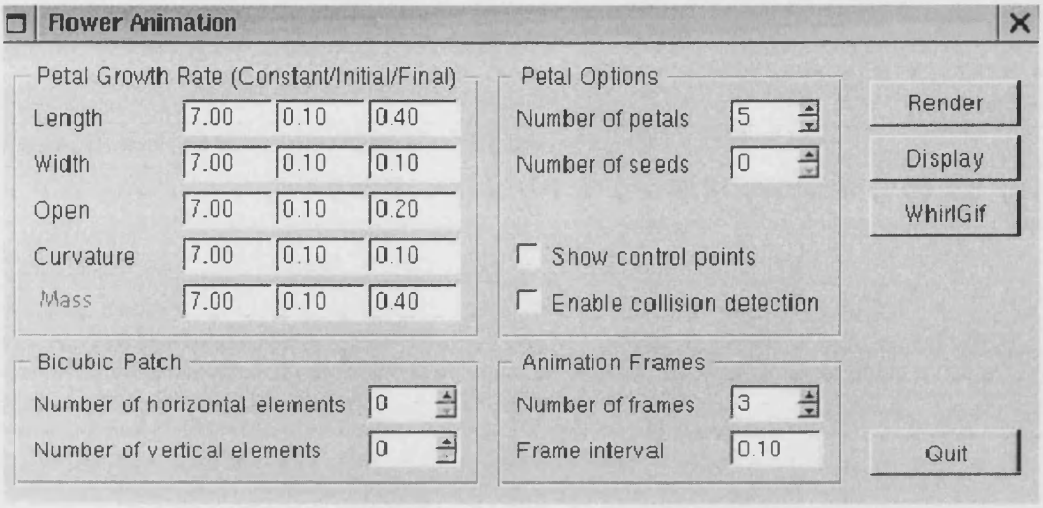



Figure A.1: User interface

- 'Petal growth rates' section provides the parameters related to the relevant growth function. The initial and final factors are the minimum and maximum value of the growth. The 'constant' is the growth change rate. It is easy to understand the factors related to length, width and curvature change. Factor 'open' is the flower opening speed. And 'Mass' is the petal mass, the petal mass increases as it grows.

- 'Petal options' is for the whole flower structure design. The user can choose the numbers of petals and seeds required. The arrangement of petals and seeds will depend on the phyllotaxis technique in the system.

- 'Bicubic patch' and 'show control points' options are there to help the user understand the surface structure by showing the patches or control points.

- 'Animation frames' option allows the user to choose the number of frames and the time interval between the frames.

- The choice for 'enable collision detection' is to enable the genetic algorithm

optimisation collision avoidance function.

- Click 'render' will allow the system to render it by POV-ray. And clicking 'display' will show all the frames one by one by using XV command. WhirlGif will connect all the frames to a GIF animation file.

### A.1.3 Time step

Whereas in the real world time is continuous, so that there is no break between one instant and another, in dynamics time is treated discretely. That is, time is broken up into a fixed number of measurable increments. These increments are called time steps. The smaller the time steps, the more closely they approximate the continuous nature of time, and therefore the more accurately they simulate an animated scene. However, smaller time steps also mean more calculations. Consequently, our systems allow a user to define the size of the time steps, giving user the option of trading off accuracy of against speed of calculations.

## A.2 Forward Difference

It is necessary to determine the optimal method for moving control points such that we avoid collisions between petal surfaces. As we use two sets of four control points to manipulate the body of the petal surface, we must first generate the two Bézier curves, one for each set of control points.

A basic way to draw a parametric cubic is by interactive evaluation of $x(u)$, $y(u)$ and $z(u)$ for incrementally spaced values of $u$, and plotting lines between successive points [Foley90]. A much more efficient method for evaluating polynomial equations is to recursively generate each succeeding value of the function

by incrementing the previously calculated value for the cubic equation through the use of finite differences. Given:

$$f(u) = a_0u^3 + a_1u^2 + a_2u + a_3 \tag{A.1}$$

and

$$f_{i+1} = f_i + \Delta f_i \tag{A.2}$$

where $\Delta f_i$ is the forward difference. The function $f_i$ is evaluated at $u_i$, and $f_{i+1}$ is evaluated at $u_{i+1} = u_i + \delta$, where $\delta$ is the step size for incrementing $u$. For a cubic curve, the forward difference evaluates to

$$\Delta f_i = 3a_0\delta u_i^2 + (3a\delta^2 + 2a_1\delta)u_i + a_0\delta^3 + a_1\delta^2 + a_2\delta \tag{A.3}$$

which is a quadratic function of $u$. However, we can use the same incremental procedure to obtain successive values of $\Delta f$. That is ,

$$\Delta f_{i+1} = \Delta f_i + \Delta^2 f_i \tag{A.4}$$

where the second forward difference is a linear function of $u$:

$$\Delta^2 f_i = 6a_0\delta^2 u_i + 6a_0\delta^3 + 2a_1\delta^2 \tag{A.5}$$

Repeating this process once more, we can write

$$\Delta^2 f_{i+1} = \Delta^2 f_i + \Delta^3 f_i \tag{A.6}$$

with the third forward difference as the constant

$$\Delta^3 f_i = 6a_0\delta^3 \tag{A.7}$$

and

$$\Delta^4 f_i = \Delta^5 f_i = \cdots = \Delta^n f_i \equiv 0 \tag{A.8}$$

Therefore only equations (2), (4), (6), and (7) are needed to incrementally obtain points along the complete curve from $u = 0$ to $u = 1$ with a step size $\delta$. The initial values at $i = 0$ and $u = 0$ are:

$$f_0 = a_3 \tag{A.9}$$

$$\Delta f_0 = a_0\delta^3 + a_1\delta^2 + a_2\delta \tag{A.10}$$

$$\Delta^2 f_0 = 6a_0\delta^3 + 2a_1\delta^2 \tag{A.11}$$

Calculations for successive points are then efficiently carried out as a series of additions. To apply this incremental procedure to Bézier curves, three sets of calculations are needed for the coordinates $x(u)$, $y(u)$, and $z(u)$. For surfaces, incremental calculations are applied for both values of $u$ and $v$.

## A.3 Genetic Algorithms Implementation

The main genetic algorithms implementations are:

```
//start with an initial time
t = 0;
```

```
//initialise a usually random population of individuals
initpopulation P(t);

//evaluate fitness of all initial individuals of population
evaluate P(t);

//test for termination criteria (time, fitness, etc.)
while not done do {

        //increase the time counter
        t = t + 1;

        //select a sub-population for offspring production
        P' = selectparents P(t);

        //genetic operation crossover
        crossover P'(t);

        //genetic operation mutation
        mutation P'(t);

        //evaluate it's new fitness
        evaluate P'(t);

        //select the survivors from actual fitness
```

```
        P = survive (P, P'(t));
}
```

# Bibliography

[Aono84a] Aono,M, Kunii,T: Botanical Tree Image Generation, *IEEE Computer Graphics and Applications*, Vol. 4, No. 5, pp.10-34, 1984.

[Arai96a] Arai,K, Kurihara,T, Anjyo,K: Bilinear Interpolation for Facial Expression and Metamorphosis in Real-time Animation, *Visual Computer*, pp.105,1996.

[Barte87a] Bartels,R, Beatty,J, Barsky,B: *An Introduction to Splines for use in Computer Graphics & Geometrics Modeling, Morgan Kaufmann Publishers*, 1987.

[Bell91] Bell,A: *Plant Form: An Illustrated Guide to Flowering Plant Morphology*, 1991.

[Benso79] Benson,L: *Plant Classification*,1979.

[Borre94a] Borrel,P, Rappoport,A: Simple Constrained Deformations for Geometric Modeling and Interactive Design, *ACM Trans. on Graphics*, pp.137, 1994.

[Bruce96a] Bruce,V, Green,P, Georgeson,M: *Visual Perception,Psychology Press*, 1996.

[Brunn97] Brunnstrom,K, Stoddart,A: Genetic Algorithms for Free-Form Surface Matching, *13th International Conference on Pattern Recognition*, 1997.

[Chiba94a] Chiba,N, Ohshida,K, Muraoka,K, Miura,M: A Growth Model Having the Abilities of Growth-regulations for Simulating Visual Nature of Botanical Trees, *Computers & Graphics*,Vol. 18, No. 4, pp.469-479, 1994.

[Chiba96b] Chiba,N, Ohshida,K: Visual Simulation of Leaf Arrangement and Autumn Colours, *The Journal of Visualization and Computer Animaiton*, Vol. 7, pp.79-93, 1996.

[Demko85a] Demko,S, Hodges,L, Naylor,B: Construction of Fractal Objects with Iterated Function Systems, *SIGGRAPH*, pp.271-278, 1985.

[DeRos98] DeRose,T, Kass,M, Truong,T: Subdivision Surfaces in Character Animation, *SIGGRAPH*, pp.85-94, 1998.

[DeRos93] DeRose,T, Goldman,R, Hagen,H, Mann,S: Functional Composition Algorithms via Blossoming, *ACM Transactions on Graphics*, Vol. 12, No. 2, pp.113-135, April 1993.

[Dobas00] Dobashi,Y, Kaneda,K, Yamashita,H, Nishita,T: A Simple, Efficient Method for Realistic Animation of Clouds, *SIGGRAPH*, 2000.

[Doug99] Doug,J, Dinesh,P: Accurate Real Time Deformable Objects, *SIGGRAPH*, 1999.

[Durik98a] Durikovic,R, Kaneda,K, Yamashita,H: Animation of Biological Organ Growth Based on L-systems, *Computer Graphics Forum*, Vol. 17, No. 3, 1998.

[Edels88] Edelstein-Keshet,L: *Mathematical Models in Biology,* Random House, New York, 1988.

[Farin88a] Farin,G: *Curves and Surfaces for Computer Aided Geometric Design, Academic Press*, 1988.

[Fireb93a] Firebaugh,M: *Computer Graphics Tools for Visualization, Wm.C.Brown*, 1993.

[Foley90] Foley,J, Dam,A, Feiner,S, Hughes,J: *Computer Graphics: principles and practice*, second edition, 1990.

[Fowle92a] Fowler,D, Prusinkiewicz,P, Battjes,J: A Collision-based Model of Spiral Phyllotaxis, *Computer Graphics,* Vol. 26, No. 2, pp.361-368, July 1992.

[Goldb89] Goldberg,D: *Genetic Algorithm in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.

[Guo97a] Guo,B: Surface Reconstruction: from Points to Splines, *CAD*, pp.269, 1997.

[Hanan92a] Hanan,J: *Parametric L-systems and their application to the modelling and visualization of plants, PhD thesis, University of Regina*, June 1992.

[Holm79] Holm,E: *The Biology of Flowers*, Penguin Books Ltd, 1979.

[Holto94a] Holton,M: Strands, Gravity and Botanical Tree Imagery, *Computer Graphics forum,* Vol. 13, No. 1, pp.57-67, 1994.

[Hunt78a] Hunt,R: *Plant Growth Analysis, Arnold*, 1978.

[Jacob79a] Jacobs,W: *Plant Hornones and Plant Development, Cambridge Univ Press*, 1979.

[Juttl98] Juttler,B: Convex Surface Fitting with Parametric Bézier Surfaces, *Mathematical Methods for Curves and Surfaces II*, pp263-270, 1998.

[Krish97] Krishnan,S, Manocha,D: An Efficient Surface Intersection Algorithm Based on Lower-Dimensional Formulation,*ACM Transactions on Graphics*, Vol. 16 , No. 1 , pp.76-106, January 1997.

[Leon97a] Leon,L, Veron,P: Semiglobal Deformation and Correction of Free-Form Surfaces Using a Mechanical Alternative, *Visual Computer*, pp.109, 1997.

[Linde76] Lindenmayer,A, Rozemberg,G: Automata, Languages, Development, 1976.

[Linte99a] Lintermann,B, Deussen,O: Interactive Modeling of Plants, *IEEE Computer Graphics and Applications*, pp.56-65, Jan/Feb, 1999.

[Liu94] Liu,Z, Gortler,S, Cohen,M: Hierarchical spacetime control, *SIGGRAPH*, pp.35-42, 1994.

[Loomi97] Loomis,J, Liu, X, Ding,Z, Fujimura,K, Evans,M, Ishikawa,H: Visualization of plant growth. *IEEE Visualization '97*, 1997.

[Lu00] Lu,Z, Willis,C, Paddon,D: Perceptually Realistic Flower Generation, *WSCG*, Plzen, Czech Republic, 2000.

[Lu01] Lu,Z, Willis,C, Paddon,D: Surface Animation for Flower Growth, *Mathematical Methods for Curves and Surfaces: Oslo 2000*, Vanderbilt University Press, 2001.

[MacCr96] MacCracken,R, Joy,K: Free-form Deformations with Lattices of Arbitrary Topology, *SIGGRAPH*, pp.181-188, 1996.

[Mech96a] Mech,R, Prusinkiewicz,P: Visual Models of Plants Interacting with Their Environment, *Proceedings of SIGGRAPH* , pp.397-410, 1996.

[Murch73a] Murch,G: *Visual and Auditory Perception, The Bobbs-Merrill Company*, 1973.

[Ngo93] Ngo,J, Marks,J: Spacetime constraints revisited, *SIGGRAPH*, pp.343-350, 1993.

[Provo95] Provot,X: Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behavior, *Proc. Graphics Interface*, 1995.

[Prusi90a] Prusinkiewicz,P, Lindenmayer,A, etc: *The Algorithmic Beauty of Plants, Springer-Verlag*, 1990.

[Prusi93b] Prusinkiewicz,P, Hammel,M, Mjolsness,E: Animation of plant development, *SIGGRAPH*, pp.351-360, 1993.

[Rana95] Rana,A, Zalzala,A: An Evolutionary Algorithm for COllision Free Motion Planning of Multi-arm Robots, *Proceeding of Genetic Algorithms in Engineering Systems: Innovations and Applicaitons*, pp.123-130, 1995.

[Reyno87] Reynolds,C: Flocks, Herds, and Schools: A Distributed Behavioral Model, *SIGGRAPH*, pp.25-34, 1987.

[Seder95] Sederberg,T, Chen,F: Implicitization using Moving Curves and Surfaces, *SIGGRAPH*, 1995.

[Sekul94a] Sekuler,R, Blake,R: *Perception, McGraw-Hill*, 1994.

[Sims94] Sims,K: Evolving Virtual Creatures, *SIGGRAPH*, pp.15-22, 1994.

[Skala97] Skalak,R, Farrow,D, Hoger,A: Kinematics of Surface Growth, *Journal of Mathematical Biology,* Vol. 35, pp.869-907, 1997.

[Stewa95] Stewart,I: *Nature's Numbers.* Phoenix, 1995.

[Thalm90a] Thalmann,D: *Scientific Visualization and Graphics Simulation, WILEY*,1990.

[Thorn76a] Thornley,J: *Mathematical Models in Plant Physiology,* Academic Press, 1976.

[Terzo87] Terzopoulos,D, Platt,J, Barr,A, Fleischer,K: Elastically Deformable Models, *SIGGRAPH*, 1987.

[Terzo88] Terzopoulos,D, Fleischer,K: Modeling Inelastic Deformation: Viscoelasticity, Plasticity, Fracture, *SIGGRAPH*, 1988.

[Tu94] Tu,X, Terzopoulos,D: Artificial Fishes: Physics, Locomotion, Perception, Behavior, *SIGGRAPH*, 1994.

[Ugail99] Ugail,H, Bloor,M, Wilson,M: Techniques for Interactive Design Using the PDE method, *ACM Transactions on Graphics*, 18(2), pp195-212, 1999.

[Vienn89a] Viennot,X, Eyrolles,G, Janey,N, Arques,D: Combinatorial Analysis of Ramified Patterns and Computer Imagery of Trees, *SIGGRAPH*, pp.31-40, 1989.

[Vogel79] Vogel,H: A Better way to construct the sunflower head. *Mathematical Biosciences,* Vol 44,pp.179-189,1979.

[Weber95a] Weber,J, Penn,J: Creation and Rendering of Realistic Trees, *Proceedings of SIGGRAPH* , pp.119-128, 1995.

[Welch92a] Welch,W, Witkin,A: Variational Surface Modeling, *SIGGRAPH*, 1992.

[Zakar01] Zakaria,M: lnteractive Evolutionary Approach to Character Animation, *WSCG*, 2001.